

AD-A060 684

MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 1967-76-C-0187

UNCLASSIFIED

ESD-TR-78-129-VOL-2

NL

1 OF 5

AD
A060684

LEVEL III

Vol. I

A060683

USER REQUIREMENTS LANGUAGE (URL)
USER'S MANUAL PART II (REFERENCE)
IBM/370/MVS/TSO/VERSION 3.2

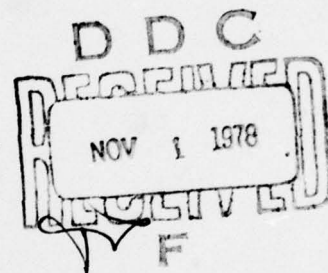


AD A060684

DDC FILE COPY

ISDOS Project
University of Michigan
Department of Industrial and Operations Engineering
Ann Arbor, Michigan 48109

March 1977



Approved for Public Release;
Distribution Unlimited.

Prepared for

DEPUTY FOR TECHNICAL OPERATIONS
ELECTRONIC SYSTEMS DIVISION
HANSCOM AIR FORCE BASE, MA 01731

78 10 23 050

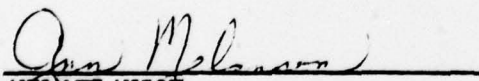
LEGAL NOTICE

When U. S. Government drawings, specifications or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

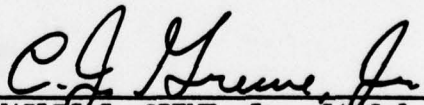
OTHER NOTICES

Do not return this copy. Retain or destroy.

This Technical Report has been reviewed and is approved for publication.

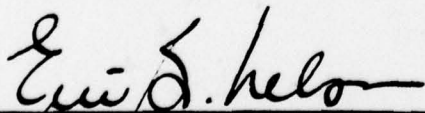


ANN MELANSON
Project Engineer



CHARLES J. GREWE, Jr., Lt Col, USAF
Chief, Technology Applications Division

FOR THE COMMANDER



ERIC B. NELSON, Colonel, USAF
Acting Director, Computer Systems Engineering
Deputy for Technical Operations

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER 18 ESD/TR-78-129-VOL 2-2	2. GOVT ACCESSION NO.	3. REPORT TYPE AND PERIOD COVERED 9 Technical rept.	4. REPORT DATE 11 March 77
5. TITLE (and Subtitle) 6 USER REQUIREMENTS LANGUAGE (URL) User's Manual, Part II. (Reference) IBM/370/MVS/TSC/Version 3.2.		6. PERFORMING ORG. REPORT NUMBER CDRL Item 004	7. CONTRACT OR GRANT NUMBER(s) 15 F19628-76-C-0197
7. AUTHOR(s) ISDOS Project		8. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 16 PE64740F, Project 2237	
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Michigan Dept. of Industrial & Operations Engineering Ann Arbor, Michigan 48109		10. CONTROLLING OFFICE NAME AND ADDRESS 11 Deputy for Technical Operations Electronic Systems Division Hanscom AFB, MA 01731	
11. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 10 451P		12. SECURITY CLASS. (of this report) UNCLASSIFIED	
13. DISTRIBUTION STATEMENT (of this Report)		14. DECLASSIFICATION/DOWNGRADING SCHEDULE	
15. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
16. SUPPLEMENTARY NOTES			
17. KEY WORDS (Continue on reverse side if necessary and identify by block number)			
Computer-Aided Design Information Processing Information System Requirements Requirements Analysis		Requirements Language Requirements Specification Specification Analysis	
18. ABSTRACT (Continue on reverse side if necessary and identify by block number)			
This report is part of a series that deals with a Computer-Aided Design and Specification Analysis Tool (CADSAT). The purpose of the tool is to describe the requirements for information processing systems and to record such descriptions in machine-processable form. The major components of CADSAT are the User Requirements Language (URL) and the User Requirements Analyzer (URA) which can operate in an interactive computer environment. This report, Part I and Part II describes how the formal URL may be used to define systems. It explains the language statements available, their use and application on an IBM/370/MVS/TSO computer.			

URL Language Reference Manual

PREFACE

↓ This manual describes Version 3.2 of the User Requirements Language (URL).

The manual consists of two parts. Part II is a reference manual, which shows the proper syntax for each statement, and is intended only for reference use by those who have read Part I and/or received formal instruction in URL. Refer to Part I for a detailed description of the language statements available and their use.

↙

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
J.S. : RATION	
DISTRIBUTION/AVAILABILITY CODES	
SPECIAL	
A	

- B -

URL Language Reference Manual

CONTENTS

1.0	INTRODUCTION	1
2.0	THE LANGUAGE	2
2.1	Introduction	2
2.2	URL Character Set	3
2.3	Words	3
2.4	Integer	3
2.5	Names	3
2.6	Punctuation	4
2.7	Name(s)	4
2.8	Statement Formation	4
2.9	Sections	4
2.10	Comment-entry	4
2.11	Comments	4
2.12	Notation Used in Describing Syntax	5
3.0	SECTION SUMMARIES	6
3.1	Statements Allowed in Most Sections ...	6
3.2	CONDITION section	8
3.3	DEFINE section	10
3.4	DESIGNATE section	12
3.5	ELEMENT section	13
3.6	ENTITY section	15
3.7	EVENT section	17
3.8	GROUP section	19
3.9	INPUT section	21
3.10	INTERFACE section	23
3.11	INTERVAL section	25
3.12	MEMO section	26
3.13	OUTPUT section	27
3.14	PROBLEM-DEFINER section	29
3.15	PROCESS section	30
3.16	PROCESSOR section	34
3.17	RELATION section	36
3.18	RESOURCE section	39
3.19	RESOURCE-USAGE-PARAMETER section	40
3.20	SET section	41
3.21	UNIT section	43
4.0	INDIVIDUAL STATEMENTS by SECTION	44
4.1	CONDITION SECTION	45
	ASSERT	46
	ATTRIBUTES	47
	BECOMING CAUSES	48
	BECOMING INTERRUPTS	49
	BECOMING TERMINATES	50
	BECOMING TRIGGERS	51
	DESCRIPTION	52
	KEYWORDS	53
	MADE	54
	RESPONSIBLE-PROBLEM-DEFINER	55

URL Language Reference Manual

SECURITY	56
SEE-MEMO	57
SOURCE	58
SYNONYMS	59
TRACE-KEY	60
WHILE	61
 4.2 DEFINE SECTION	63
APPLIES	64
ASSERT	65
ATTRIBUTES	66
DESCRIPTION	67
KEYWORDS	68
MAINTAINED	69
RESPONSIBLE-PROBLEM-DEFINER	70
SECURITY	71
SEE-MEMO	72
SOURCE	73
SUBSETTING-CRITERION	74
SYNONYMS	75
TRACE-KEY	76
VALUES	77
 4.3 DESIGNATE SECTION	78
 4.4 ELEMENT SECTION	79
ASSERT	80
ASSOCIATED	81
ATTRIBUTES	82
CLASSIFICATION	83
CONTAINED	84
DERIVED	85
DESCRIPTION	86
IDENTIFIES	87
KEYWORDS	88
RESPONSIBLE-PROBLEM-DEFINER	89
SECURITY	90
SEE-MEMO	91
SOURCE	92
SUBSETTING-CRITERION	93
SYNONYMS	94
TRACE-KEY	95
UPDATED	96
USED	97
VALUES	98
 4.5 ENTITY SECTION	99
ASSERT	100
ATTRIBUTES	101
CARDINALITY	102
CLASSIFICATION	103
CONSISTS	104
CONTAINED	105
DERIVED	106
DESCRIPTION	107
IDENTIFIED	108

URL Language Reference Manual

KEYWORDS	109
RELATED	110
RESPONSIBLE-PROBLEM-DEFINER	111
SECURITY	112
SEE-MEMO	113
SOURCE	114
SYNONYMS	115
TRACE-KEY	116
UPDATED	117
USED	118
VOLATILITY	119
 4.6 EVENT SECTION	 120
ASSERT	121
ATTRIBUTES	122
CAUSED	123
CAUSES	124
DESCRIPTION	125
HAPPENS	126
INCEPTION	127
INTERRUPTS	128
KEYWORDS	129
MAKES	130
RESPONSIBLE-PROBLEM-DEFINER	131
SECURITY	132
SEE-MEMO	133
SOURCE	134
SYNONYMS	135
TERMINATES	136
TERMINATION	137
TRACE-KEY	138
TRIGGERS	139
 4.7 GROUP SECTION	 140
ASSERT	141
ASSOCIATED	142
ATTRIBUTES	143
CLASSIFICATION	144
CONSISTS	145
CONTAINED	146
DERIVED	147
DESCRIPTION	148
IDENTIFIES	149
KEYWORDS	150
RESPONSIBLE-PROBLEM-DEFINER	151
SECURITY	152
SEE-MEMO	153
SOURCE	154
SUBSETTING-CRITERION	155
SYNONYMS	156
TRACE-KEY	157
UPDATED	158
USED	159
 4.8 INPUT SECTION	 160
ASSERT	161

URL Language Reference Manual

ATTRIBUTES	162
CAUSES	163
CLASSIFICATION	164
CONSISTS	165
CONTAINED	166
DESCRIPTION	167
GENERATED	168
HAPPENS	169
INTERRUPTS	170
KEYWORDS	171
MAKES	172
PART	173
RECEIVED	174
RESPONSIBLE-PROBLEM-DEFINER	175
SECURITY	176
SEE-MEMO	177
SOURCE	178
SUBPARTS	179
SYNONYMS	180
TERMINATES	181
TRACE-KEY	182
TRIGGERS	183
USED	184
 4.9 INTERFACE SECTION	 185
ASSERT	186
ATTRIBUTES	187
DESCRIPTION	188
GENERATES	189
KEYWORDS	190
PART	191
RECEIVES	192
RESPONSIBLE	193
RESPONSIBLE-PROBLEM-DEFINER	194
SECURITY	195
SECURITY-ACCESS-RIGHT	196
SEE-MEMO	197
SOURCE	198
SUBPARTS	199
SYNONYMS	200
TRACE-KEY	201
 4.10 INTERVAL SECTION	 202
ASSERT	203
ATTRIBUTES	204
CONSISTS	205
DESCRIPTION	206
KEYWORDS	207
RESPONSIBLE-PROBLEM-DEFINER	208
SECURITY	209
SEE-MEMO	210
SOURCE	211
SYNONYMS	212
TRACE-KEY	213
 4.11 MEMO SECTION	 214

URL Language Reference Manual

APPLIES	215
ASSERT	216
ATTRIBUTES	217
DESCRIPTION	218
KEYWORDS	219
RESPONSIBLE-PROBLEM-DEFINER	220
SECURITY	221
SOURCE	222
SYNONYMS	223
TRACE-KEY	224
4.12 OUTPUT SECTION	225
ASSERT	226
ATTRIBUTES	227
CLASSIFICATION	228
CONSISTS	229
CONTAINED	230
DERIVED	231
DESCRIPTION	232
GENERATED	233
HAPPENS	234
KEYWORDS	235
PART	236
RECEIVED	237
RESPONSIBLE-PROBLEM-DEFINER	238
SECURITY	239
SEE-MEMO	240
SOURCE	241
SUBPARTS	242
SYNONYMS	243
TRACE-KEY	244
4.13 PROBLEM-DEFINER SECTION	245
ASSERT	246
ATTRIBUTES	247
DESCRIPTION	248
KEYWORDS	249
MAILBOX	250
RESPONSIBLE	251
SECURITY	252
SEE-MEMO	253
SOURCE	254
SYNONYMS	255
TRACE-KEY	256
4.14 PROCESS SECTION	257
ASSERT	258
ATTRIBUTES	259
DERIVES	260
DESCRIPTION	261
GENERATES	262
HAPPENS	263
INCEPTION-CAUSES	264
INTERRUPTED	265
INTERRUPTS	266
KEYWORDS	267

URL Language Reference Manual

MAINTAINS	268
MAKES	269
PART	270
PERFORMED	271
PROCEDURE	272
RECEIVES	273
RESPONSIBLE-PROBLEM-DEFINER	274
RESOURCE-USAGE	275
SECURITY	276
SECURITY-ACCESS-RIGHT	277
SEE-MEMO	278
SOURCE	279
SUBPARTS	280
SYNONYMS	281
TERMINATED	282
TERMINATES	283
TERMINATION-CAUSES	284
TRACE-KEY	285
TRIGGERED	286
TRIGGERS	287
UPDATES	288
USES	289
UTILIZED	290
UTILIZES	291
 4.15 PROCESSOR SECTION	 292
ASSERT	293
ATTRIBUTES	294
CONSUMES	295
DESCRIPTION	296
KEYWORDS	297
PART	298
PERFORMS	299
RESPONSIBLE-PROBLEM-DEFINER	300
SECURITY	301
SECURITY-ACCESS-RIGHT	302
SEE-MEMO	303
SOURCE	304
SUBPARTS	305
SYNONYMS	306
TRACE-KEY	307
 4.16 RELATION SECTION	 308
ASSERT	309
ASSOCIATED-DATA	310
ATTRIBUTES	311
BETWEEN	312
CARDINALITY	313
CONNECTIVITY	314
DERIVATION	315
DESCRIPTION	316
KEYWORDS	317
MAINTAINED	318
RESPONSIBLE-PROBLEM-DEFINER	319
SECURITY	320
SEE-MEMO	321

URL Language Reference Manual

SOURCE	322
SYNONYMS	323
TRACE-KEY	324
 4.17 RESOURCE SECTION	 325
ASSERT	326
ATTRIBUTES	327
CONSUMED	328
DESCRIPTION	329
KEYWORDS	330
MEASURED	331
RESPONSIBLE-PROBLEM-DEFINER	332
SECURITY	333
SEE-MEMO	334
SOURCE	335
SYNONYMS	336
TRACE-KEY	337
 4.18 RESOURCE-USAGE-PARAMETER SECTION	 338
ASSET	339
ATTRIBUTES	340
DESCRIPTION	341
KEYWORDS	342
RESOURCE-USAGE-PARAMETER-VALUE ...	343
RESPONSIBLE-PROBLEM-DEFINER	344
SECURITY	345
SEE-MEMO	346
SOURCE	347
SYNONYMS	348
TRACE-KEY	349
 4.19 SET SECTION	 350
ASSET	351
ATTRIBUTES	352
CARDINALITY	353
CLASSIFICATION	354
CONSISTS	355
DERIVATION	356
DERIVED	357
DESCRIPTION	358
KEYWORDS	359
RESPONSIBLE-PROBLEM-DEFINER	360
RESPONSIBLE-INTERFACE	361
SECURITY	362
SEE-MEMO	363
SOURCE	364
SUBSET	365
SUBSETS	366
SUBSETTING-CRITERIA	367
SYNONYMS	368
TRACE-KEY	369
UPDATED	370
USED	371
VOLATILITY-MEMBER	372
VOLATILITY-SET	373

URL Language Reference Manual

4.20 UNIT SECTION	374
ASSERT	375
ATTRIBUTES	376
DESCRIPTION	377
KEYWORDS	378
MEASURES	379
RESPONSIBLE-PROBLEM-DEFINER	380
SECURITY	381
SEE-MEMO	382
SOURCE	383
SYNONYMS	384
TRACE-KEY	385

APPENDICES:

A: Notation Used in Describing Syntax	386
E: URL Reserved Words	387-391
C: URL Optional Words	392
D: Reserved Words With Synonyms	393-395
E: Name Types	396
F: Section Types	397
G: URL Forms	398

1.0 Introduction and Purpose

The original Problem Statement Language (PSL 1.0) was designed to provide the User with an improved method of stating requirements for a target information processing system (IPS). This goal was achieved by developmental work in the ISDOS Research Project leading to PSL 2.0 and URL 3.0 and their associated Analyzers (PSA 2.0 and URA 3.0). However, as with any developmental project, continued work yields improved understanding and eventually an improved product. Such is the case for URL 3.2 and the URA 3.2.

The new URL 3.2, hereafter referred to as URL, provides the User greater flexibility, more features and greater ease of use, while still maintaining the overall goals of such a computer-aided method. Therefore, URL is designed to provide understandable communication and documentation for both men and machine by having a simple syntax for the machine while maintaining the readability for the man.

The purpose of this manual is to provide a concise description of URL syntax and give brief examples of usage.

2.0 THE LANGUAGE

2.1 Introduction

Any language which is to be processed by computer needs to be structured in some way. The User Requirements Language, although it is based on English in that it uses English words and is intended to be readable as English text, must therefore be more precise than a natural language. Just as in English, the basic unit of the language is a word. In order for the Analyzer to understand URL, it treats all words as one of two types: Reserved Words, and names. Reserved Words have a specific meaning to the Analyzer and must be spelled exactly as given in the Reserved Word List (Appendix B). Many Reserved Words have a short form which may be substituted for the Reserved Word; these short forms are also given in the Reserved Word List. Some Reserved Words are essential for the URA to interpret the meaning of a statement. Other Reserved Words are not used by the Analyzer. These Reserved Words are called Optional Words (see Appendix C). Names are assigned by the User to facilitate the description of the target system. Names must be formed according to the rules given in sections 2.3 and 2.5.

These Reserved Words and names are combined with appropriate punctuation to form statements. Punctuation must be given exactly as shown in the syntax for a statement. For example, name(s) correspond to several names separated by commas; the commas are required in name(s) between each pair of names. A special punctuation symbol, a semi-colon, is used to end a statement in URL. Just as some Reserved Words are optional and do not affect the interpretation of a statement by the Analyzer, the colon is a special punctuation which may be used without affecting the meaning of a statement.

To illustrate, the syntax for the KEYWORD statement is:

KEYWORDS ARE keyword-name(s) ;

The following statements all provide equivalent information to the analyzer:

- 1) KEYWORD KEY1, KEY2, KEY3;
- 2) THE KEYWORDS ARE: KEY1, KEY2, AND KEY3;

- KEYWORD is a required Reserved Word.
- THE, ARE and AND are Optional Reserved Words.
- KEY1, KEY2, KEY3 are names.
- The commas and semi-colon are required punctuation.
- The colon is optional punctuation.

2.2 URL character set

See PART I, Section 1.6.7.

2.3 Words

A word in URL is not more than 30 contiguous code 2 or 3 characters. (See PART I, Section 1.6.7.)

2.4 Integer

An integer in URL is composed of a series of digits without decimal point, plus or minus sign.

2.5 Names

All names in URL have a type associated with them (see Appendix E for possible types). In the format for the statements, only certain types of names are allowed in certain contexts. This is indicated in the associated usage rules.

Note: Names must begin with a letter.

Note: A name in URL is any combination of not more than thirty of the above characters.

Note: Blanks may not be used in names.

2.6 Punctuation

The following characters are used for punctuation in URL:

	space (blank)
,	comma
;	semi colon
:	colon

The following rules apply to the use of punctuation in URL:

- When any punctuation appears in the format for a statement, the punctuation must be given exactly as shown.
- Two or more blanks are treated the same as a single blank.
- Blanks may be used anywhere except in words or integers.
- A colon may be used anywhere that a blank is allowed.
- A semi-colon may only be used to end a statement.

2.7 Name(s)

Name(s) is a series of names separated by commas.

2.8 Statement formation

Statements are formed from words and punctuation according to the rules given in chapter 3 and 4.

General rules:

- All statements must end in a semi-colon.
- Words must be separated by at least one character (punctuation, blank etc.).
- Any punctuation in the format descriptions of chapter 3 or 4 must be given exactly as shown.
- All statements, except section header statements, may be preceeded by optional name(s). The names must be used in the header statement for the section in which the statement occurs. If the name(s) are not given then the statement applies to all the names in the header statement. Alternately, if the name(s) are given, the statement will apply only to names in the list.

2.9 Sections

A problem statement in URI consists of at least one section. The possible section types are given in Appendix F. A section is a series of statements the first of which is a header statement; the type of header statement determines the type of section. The other statements in a section may be given in any order.

General rules:

- Only certain types of statements are allowed in a section, depending on the section type. The specific statements allowed in any section are given in chapter 3.

2.10 Comment-entry

Several statements have a comment-entry associated with them. Comment-entries are handled by the analyzer as follows:

- The rest of the input line containing the semi-colon after the reserved word for the statement is discarded
- Lines are read and added to the data base as given, up to and including the first line which contains a semi-colon.
- The semi-colon is replaced with a blank in this line before the line is added to the data base. (Note: then complete line is added to the data base even if the semi-colon is the first character in that line.)
- Parsing of statements begins at the first character of the following line.

2.11 Comments

For increased comprehension and documentation, comments (to be differentiated from comment-entries) can be used. Every comment must begin with /* and end with characters reversed, i.e., */. No blanks or other characters may appear between these characters, they must be immediately adjacent. Comments are treated exactly as a blank and do not otherwise affect the analysis of the User Requirements. Although they appear in the URA As-Is-Source Listing, they are discarded by the analyzer and

are not entered into the data base.

2.12 Notation Used in Describing Syntax

In this manual, the following notation is used when describing URL 3.2 syntax.

Lower Case Words

Words written in lower case call for names to be made up and inserted by the User. The lower case descriptions of user defined names tell what kind of words the User is to make up.

Braces

When words or phrases are enclosed in braces ({ }), a choice among the two or more entries must be made. It is important to note that one of the options must be chosen. Several braces vertically on a page is equivalent to one large brace.

Brackets

Whenever notation in a model appears within brackets ([]), it indicates some feature the User may optionally use. Several brackets vertically on a page is equivalent to one large bracket.

Ellipsis

The ellipsis (...) signifies that the URL construct immediately preceding the ellipsis can be repeated as many times as desired by the User.

Underscoring

All upper case words which are underscored are URL Reserved Words and, if used, must appear exactly as shown.

System-Parameter

The use of system-parameter in the statement syntax denotes that the system-parameter name or integer can be used.

3.0 SECTION SUMMARIES

3.1 Statements Allowed in Most sections

The following statements are allowed in almost every section:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

<u>ATTRIBUTES</u> ARE attr-name	{ attv-name }	[{ attv-name }]
	{		,attr-name	{]
	integer	}		integer	}

...

DESCRIPTION ;
comment-entry ;

KEYWORDS ARE keyword-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

With the following exceptions:

-The RESPONSIBLE-PROBLEM-DEFINER statement is not allowed in a PROBLEM-DEFINER section.

-The SEE-MEMO statement is not allowed in the MEMO section.

-The KEYWORDS statement is not allowed in a DEFINE section for a KEYWORD.

-The SOURCE statement is not allowed in a DEFINE section for a SOURCE.

-The SECURITY statement is not allowed in a DEFINE section for a SECURITY.

-The TRACE-KEY statement is not allowed in a DEFINE section for a TRACE-KEY.

-No statements are allowed in a DESIGNATE section.

CONDITION name(s) ;

```
[, name attribute-name attribute-value] ...;
```

```

      { TRUE }
BECOMING { } CAUSES event-name(s);
      { FALSE }

```

```

      { TRUE }
BECOMING {   } INTERRUPTS process-name(s) ;
      { FALSE }

```

```

{ TRUE }
BECOMING { } TERMINATES process-name(s);
{ FALSE }

```

```

      { TRUE }
BECOMING {   } TRIGGERS process-name(s);
      { FALSE }

```

DESCRIPTION ;
comment-entry ;

KEYWORDS ARE keyword-name(s) :

```

MADE { TRUE } BY event-
      { FALSE } input-name(s) ;
      process-

```

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

3.2 CONDITION section (continued)

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

{ TRUE }

{ } WHILE ;

{ FALSE }

comment-entry ;

3.3 DEFINE section

	{ATTRIBUTE	}[{ATTRIBUTE	}]	
	{ATTRIBUTE-VALUE	}[{ATTRIBUTE-VALUE	}]	
	{CLASSIFICATION	}[{CLASSIFICATION	}]	
	{KEYWORD	}[{KEYWORD	}]	
	{MAILBOX	}[{MAILBOX	}]	
<u>DEFINE</u> name	{SECURITY	}[, name	{SECURITY	}] ... ;	
	{SOURCE	}[{SOURCE	}]	
	{SUBSETTING-CRITERION}	[{SUBSETTING-CRITERION}]	
	{SYSTEM-PARAMETER	}[{SYSTEM-PARAMETER	}]	
	{TRACE-KEY	}[{TRACE-KEY	}]	

APPLIES TO name(s) ;

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ... ;

ATTRIBUTES ARE attr-name { attrv-name } [, attr-name { attrv-name }] ...
 { integer } [{ integer }]

DESCRIPTION ;
 comment-entry ;

KEYWORDS ARE keyword-name(s) ;

MAINTAINED BY process-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SII-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SUBSETTING-CRITERION FOR set-name(s) ;

3.3 DEFINE section (continued)

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

```

      {                integer                }
      {                }
VALUES ARE { { *min    }                { *max  } } ;
             { {        } THRU          {        } }
             { { NEGINF }                { POSINF } }
```

* Min and max must be non-negative integers.

3.4 DESIGNATE section

DESIGNATE name AS A SYNONYM FOR name

[, name AS A SYNONYM FOR name] ... ;

ELEMENT names(s) ;

```
[, name attribute-name attribute-value] ...;
```

```

ATTRIBUTES ARE attr-name { attr-name } [
                                { attr-name } ]
                                { integer } [ ,attr-name { attr-name } ] ..
                                { integer } [ { integer } ]

```

```
[, classification-name [ integer ])... ;
```

CONTAINED IN group-
 entity-
 input-name(s) ;
 out put-

```

[          group-          ]
[          entity-        ]
DERIVED BY process-name(s) [ USING      set-name(s) ] ;
[          input-         ]
[          element-       ]

```

DESCRIPTION ;
comment-entry ;

IDENTIFIES entity-name(s) :

KEYWORDS ARE keyword-name(s) ;

3.5 ELEMENT section (continued)

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SIE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SUBSETTING-CRITERION FOR set-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

UPDATED BY process-name(s) [USING [group- entity- element- input- set-]] ;

USED BY process-name(s) [TO [{ DERIVE } *output- entity- name(s)] ; [{ UPDATE } group- element-]]

* Output-name(s) may only be used with the DERIVE clause.

VALUES ARE { integer }
 { { *min } { *max } } ;
 { { } THRU { } }
 { { NEGINF } { POSINF } }

* Min and max must be integers.

3.6 ENTITY section

ENTITY name(s) ;

ASSERT name attribute-name attribute-value
 [, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [
 { integer } [,attr-name { attrv-name }] ..
 { integer } [{ integer }]] ..

CARDINALITY IS system-parameter ;

CLASSIFICATION classification-name [integer]
 [, classification-name [integer]]... ;

CONSISTS OF [system-parameter] element-
 group-name
 element-
 [, [system-parameter] group-name] ... ;

CONTAINED IN set-name(s) ;

DERIVED BY process-name(s) [USING group-
 entity-
 set-name(s)] ;
 [input-
 element-]

DESCRIPTION ;
 comment-entry ;

IDENTIFIED BY group
 element-name(s) ;

KEYWORDS ARE keyword-name(s) ;

3.6 ENTITY section (continued)

RELATED TO entity-name VIA relation-name ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

	[group-]
	[entity-]
<u>UPDATED</u> BY process-name(s)	[<u>USING</u> element-	name(s)] ;
	[input-]
	[set-]

	[set-]
	[{ <u>DERIVE</u> }	*output-
<u>USED</u> BY process-name(s)	[TO { }	entity-
	[{ <u>UPDATE</u> }	group-
	[element-]

* Output-name(s) may only be used with the DERIVE clause.

VOLATILITY ;
comment-entry ;

EVENT name(s) :

```
[, name attribute-name attribute-value] ...;
```

```

ATTRIBUTES ARE attr-name { attv-name } [ { attv-name } ]
                                { integer } [ ,attr-name { integer } ] ..
                                { integer } [ { integer } ]

```

CAUSED BY event-
name(s) ;
input-

```

CAUSED WHEN condition-name BECOMES { TRUE }
                                     { FALSE };

```

DESCRIPTION ;
comment-entry ;

CN INCEPTION OF process-name(s) :

KEYWORDS ARE keyword-name(s) :

```

      { TRUE }
MAKES condition-name(s) { FALSE } ;

```

3.7 EVENT section (continued)

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TERMINATES process-name(s) ;

ON TERMINATION OF process-name(s) ;

TRACE-KEY trace-key-name(s) ;

TRIGGERS process-name(s) ;

GROUP name(s) :

```
[, name attribute-name attribute-value] ...;
```

```

ATTRIBUTES ARE attr-name { attr-name } [
                                { attr-name } ]
                                { integer } [
                                { integer } ] .

```

```
[, classification-name [ integer ] ]... ;
```

```
[ , [ system-parameter ] element-  
group-name ] ... ;
```

CONTAINED IN group-
 entity-
 - name(s) ;
 input-
 output-

```

[                group-                ]
[                entity-                ]
DERIVED BY process-name(s) [ USING      set-name(s) ] ;
[                input-                 ]
[                element-                ]

```

IDENTIFIES entity-name(s) ;

3.8 GROUP section (continued)

KEYWORDS ARE keyword-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SIE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SUBSETTING-CRITERION FOR set-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

UPDATED BY process-name(s) [USING [group-
entity-
element- name(s)]
input-
set-] ;

USED BY process-name(s) [TO [{ DERIVE } *output-
entity- name(s)]
[{ UPDATE } group-
element-]] ;

* Output-name(s) may only be used with the DERIVE clause.

3.9 INFUT section

INPUT name(s) ;

ASSERT name attribute-name attribute-value
 [, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [{ attrv-name }]
 { integer } [,attr-name { attrv-name }] ..
 { integer } [{ integer }]

CAUSES event-name(s) ;

CLASSIFICATION classification-name [integer]
 [, classification-name [integer]].... ;

CONSISTS OF [system-parameter] element-
 group-name
 [, [system-parameter] element-
 group-name] ... ;

CONTAINED IN set-name(s) ;

DESCRIPTION ;
 comment-entry ;

GENERATED BY interface-name(s) ;

HAPPENS system-parameter TIMES-PER interval-name ;

INTERRUPTS process-name(s) ;

KEYWORDS ARE keyword-name(s) ;

3.9 INFUT section (continued)

MAKES condition-name(s) { TRUE }
 { FALSE } ;

PART OF input-name ;

RECEIVED BY process-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SUBPARTS ARE input-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TERMINATES process-name(s) ;

TRACE-KEY trace-key-name(s) ;

TRIGGERS process-name(s) ;

	[set-]
<u>USED</u> BY process-name(s)	[{ <u>DERIVE</u> }	*output-]
	TO	{	entity-	name(s)] ;
	[{ <u>UPDATE</u> }	group-]
	[element-]

* Output-name(s) may only be used with the DERIVE clause.

SUBPARTS ARE interface-name(s) :

3.10 INTERFACE section (continued)

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

3.11 INTERVAL section

INTERVAL name (s) ;

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [{ attrv-name }]
 { integer } [, attr-name { integer }] ..

CONSISTS OF [system-parameter] interval-name

[, [system-parameter] interval-name] ... ;

DESCRIPTION ;

comment-entry ;

KEYWORDS ARE keyword-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

3.12 MEMO section

MEMO name(s) ;

APPLIES TO non-memo-name(s) ;

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [{ attrv-name }]
 { integer } [,attr-name { integer }] ;

DESCRIPTION ;

comment-entry ;

KEYWORDS ARE keyword-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

OUTPUT name(s) ;

```
[, name attribute-name attribute-value] ...;
```

CLASSIFICATION classification-name [integer]

CONSISTS OF [system-parameter] element-group-name

CONTAINED IN set-name(s) ;

DESCRIPTION ;
comment-entry ;

HAPPENS system-parameter TIMES-PER interval-name ;

PART OF output-name :

3.13 OUTPUT section (continued)

RECEIVED BY interface-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SUBPARTS ARE output-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

3.14 PROBLEM-DEFINER section

PROBLEM-DEFINER name(s) ;

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attr-name } [,attr-name { attr-name }]
 { integer } [{ integer }] .

DESCRIPTION ;
 comment-entry ;

KEYWORDS ARE keyword-name(s) ;

MAILBOX IS mailbox-name ;

RESPONSIBLE FOR name(s) ;

SECURITY IS security-name(s) ;

SFE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

3.15 PROCESS section

PROCESS name(s) ;

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

<u>ATTRIBUTES</u> ARE attr-name	{ attrv-name }	[{ attrv-name }]
	{ integer }	[,attr-name	{ integer }
]]
]

<u>DERIVES</u>	set-	[set-]
	output-	[input-]
	element-name(s)	[<u>USING</u> element-name(s)]
	entity-	[entity-]
	group-	[group-]

DESCRIPTION ;
comment-entry ;

GENERATES output-name(s) ;

HAPPENS system-parameter TIMES-PER interval-name ;

INCEPTION-CAUSES event-name(s) ;

INTERRUPTED BY event-
input-name(s) ;
process-

INTERRUPTED WHEN condition-name BECOMES { TRUE }
{ FALSE } ;

INTERRUPTS process-name(s) ;

3.15 PROCESS section (continued)

KEYWORDS ARE keyword-name(s) ;

relation-
MAINTAINS subsetting-criteria-name(s) ;

MAKES condition-name(s) { TRUE }
 { FALSE } ;

PART OF process-name ;

PERFORMED BY processor-name ;

PROCEDURE ;
 comment-entry ;

RECEIVES input-name(s) ;

RESOURCE-USAGE :
 system-parameter FOR resource-usage-parameter-name;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SECURITY-ACCESS-RIGHT classification-name [integer]
 [, classification-name [integer]]... ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

3.15 PROCESS section (continued)

SUBPARTS ARE process-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TERMINATED BY event-
 input-name(s) ;
 process-

TERMINATED WHEN condition-name BECOMES { TRUE }
 { } ;
 { FALSE }

TERMINATES process-name(s) ;

TERMINATION-CAUSES event-name(s) ;

TRACE-KEY trace-key-name(s) ;

TRIGGERED BY event-
 input-name(s) ;
 process-

TRIGGERED WHEN condition-name BECOMES { TRUE }
 { } ;
 { FALSE }

TRIGGERS process-name(s) ;

	group-	[group-]
	entity-	[entity-]
<u>UPDATES</u>	element-name(s)	[<u>USING</u> element-	name(s)] ;
	set-	[set-]
		[input-]

3.15 PROCESS section (continued)

	set-	[set-]
	input-	[{ <u>DERIVE</u> }	*output-]
<u>USES</u>	element-name(s)	[TO	{	element-	name(s)] ;
	group-	[{ <u>UPDATE</u> }	group-]
	entity-	[entity-]

* Output-name(s) may only be used with the DERIVE clause.

UTILIZED BY process-name(s) ;

UTILIZES process-name(s) ;

3.16 PROCESSOR section

PROCESSOR processor-name(s) ;

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [{ attrv-name }]
 { integer } [,attr-name { integer }] ...

CONSUMES resource-name AT RATE OF

system-parameter PER resource-usage-parameter-name;

DESCRIPTION ;
 comment-entry ;

KEYWORDS ARE keyword-name(s) ;

PART OF processor-name;

PERFORMS process-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SECURITY-ACCESS-RIGHT classification-name [integer]
 [, classification-name [integer]]... ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

3.16 PROCESSOR section (continued)

SUBPARTS ARE processor-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

3.17 RELATION section

RELATION name(s) ;

ASSERT name attribute-name attribute-value
 [, name attribute-name attribute-value] ...;

ASSOCIATED-DATA IS ^{group-}element-name(s) ;

ATTRIBUTES ARE attr-name { attrv-name } [,attr-name { attrv-name }] ...
 { integer } [{ integer }]

BETWEEN entity-name AND entity-name ;

CARDINALITY IS system-parameter ;

CONNECTIVITY IS system-parameter TO system-parameter ;

DERIVATION ;
 comment-entry ;

DESCRIPTION ;
 comment-entry ;

KEYWORDS ARE keyword-name(s) ;

MAINTAINED BY process-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SCURCE IS source-name(s) ;

3.17 RELATION section (continued)

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

3.18 RESOURCE section

RESOURCE resource-name(s);

ASSERT name attribute-name attribute-value
 [, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [
 { integer } [,attr-name { attrv-name }] ..
 { integer } [{ integer }]]

CONSUMED BY processor-name(s) AT RATE OF
 system-parameter PER resource-usage-parameter-name;

DESCRIPTION ;
 comment-entry ;

KEYWORDS ARE keyword-name(s) ;

MEASURED IN unit-name;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SFE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

TRACE-KEY trace-key-name (s) ;

SET name (s) :

```
[, name attribute-name attribute-value] ...:
```

```

ATTRIBUTES ARE attr-name { attr-name } [ { attr-name } ]
                        { integer } [ ,attr-name { integer } ] .
                        { integer } [ { integer } ]

```

```
[, classification-name [ integer ] ]... ;
```

```

                                input-
[ , [ system-parameter ] output-name ] ... ;
                                entity-

```

```

[ group- ]
[ entity- ]
DERIVED BY process-name(s) [ USING set-name(s) ] ;
[ input- ]
[ element- ]

```

RESPONSIBLE-INTERFACE IS interface-name(s) ;

3.20 SFT section (continued)

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SUBSET OF set-name(s) ;

SUBSETS ARE set-name(s) ;

SUBSETTING-CRITERIA ARE ^{group-}element-name(s) ;
subsetting-criterion-

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

		[group-]
		[entity-]
<u>UPDATED</u> BY	process-name(s)	[<u>USING</u> element-	name(s)] ;
		[input-]
		[set-]

		[set-]
		[{ <u>DERIVE</u> }	*output-]
<u>USED</u> BY	process-name(s)	[TO {	entity-	name(s)] ;
		[{ <u>UPDATE</u> }	group-]
		[element-]

* Output-name(s) may only be used with the DERIVE clause.

VCLATILITY-MEMBER ;
comment-entry ;

VCLATILITY-SET ;
comment-entry ;

3.21 UNIT section

UNIT name(s) ;

ASSERT name attribute-name attribute-value
 [, name attribute-name attribute-value] ...;

ATTRIBUTES ARE attr-name { attrv-name } [,attr-name { attrv-name }] ..
 { integer } [{ integer }]

DESCRIPTION ;
 comment-entry ;

KEYWORDS ARE keyword-name(s) ;

MEASURES resource-name(s) ;

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

SECURITY IS security-name(s) ;

SEE-MEMO memo-name(s) ;

SOURCE IS source-name(s) ;

SYNONYMS ARE synonym-name(s) ;

TRACE-KEY trace-key-name(s) ;

4.0 INDIVIDUAL STATEMENTS

The following pages give a description of all allowable URL statements. With each statement there is a declaration of purpose, the syntax, complementary statements (if any exist), and the rules concerning the type of names allowed in the syntax and restrictions pertaining to the statement. Each page is intended to be a unit by itself; all the information needed for a statement is given on the page for that statement. Therefore, the same information may be given on several different pages.

The statements are listed alphabetically. Statements that may occur in several sections are arranged alphabetically by section type.

4.1 CONDITION section header statement

Purpose:

To indicate a TRUE/FALSE state within the system, and to optionally link that state to EVENTS and/or the initiation of PROCESSES. Thus the analyst has a way to indicate a processing path to be followed when one or more CONDITIONS are satisfied, or alternative processing paths when CONDITIONS are not met.

Syntax:

CCNDITION condition-name(s) ;

Usage Rules:

- Must be the first statement in a CONDITION section.
- More than one CONDITION can be defined at a time.

Synonyms:

CCND CONDITIONS

Examples:

- CONDITION PAYCHECK-DISTRIBUTED;

ASSERT statement

CONDITION section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value
[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

CONDITION section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```

ATTRIBUTES ARE attr-name { attrv-name } [
                                { integer } [ ,attr-name { attrv-name } ] ..
                                { integer } [ { integer } ] ]

```

Complementary Statements:
none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

BECOMING CAUSES statement

CONDITION section

Purpose:

To specify the EVENT(S) caused by this CONDITION.

Syntax:

```
      { TRUE  }  
BECOMING {     } CAUSES event-name(s) ;  
      { FALSE }
```

Complementary Statements:

CAUSED statement in the EVENT section.

Usage Rules:

- A CONDITION BECOMING TRUE or FALSE may CAUSE several different EVENTS.
- A CONDITION BECOMING TRUE may CAUSE one set of EVENTS and BECOMING FALSE may CAUSE a second set.

Synonyms:

```
{ BEC  }  
{     } CSS  
{ BECG }
```

Examples:

- BECOMING FALSE CAUSES ERROR-DETECTED ;
- BECOMING TRUE CAUSES SUBPROCESS-COMPLETION, MAIN-PROCESS-COMPLETION ;
- BEC T CSS EVENT-1, EVENT-2, EVENT-3 ;
- BECG F CSS TIME-CARD-RECOGNIZED ;

BECOMING INTERRUPTS statement

CONDITION section

Purpose:

To specify the PROCESS(ES) interrupted by a change of state for this CONDITION.

Syntax:

```
BECOMING { TRUE }  
          { FALSE } INTERRUPTS process-name(s) ;
```

Complementary Statements:

INTERRUPTED statement in the PROCESS section.

Usage Rules:

- A CONDITION BECOMING TRUE or FALSE may INTERRUPT several PROCESSES.
- A CONDITION BECOMING TRUE may INTERRUPT one set of PROCESSES and BECOMING FALSE may INTERRUPT a second set.

Synonyms:

```
{ BEC }  
{      } INTS  
{ BECG }
```

Examples:

- BECOMING FALSE INTERRUPTS NORMAL-PROCESSING ;
- BEC T INTS PACK-FOR-SHIPPING, BILLING ;
- BECG F INTS SALARY-COMPUTATION ;

BECOMING TERMINATES statement

CONDITION section

Purpose:

To specify a PROCESS/PROCESSES that are terminated when this CONDITION enters a given state.

Syntax:

```
BECOMING { TRUE }  
           { FALSE } TERMINATES process-name(s) ;
```

Complementary Statements:

TERMINATED statement in PROCESS section.

Usage Rules:

- A CONDITION BECOMING TRUE or FALSE may TERMINATE several PROCESSES.
- A CONDITION BECOMING TRUE may TERMINATE one set of PROCESSES and BECOMING FALSE may TERMINATE a second set.

Synonyms:

```
{ BEC }  
{      } TRMS  
{ BECG }
```

Examples:

- BECOMING TRUE TERMINATES BILLING-PROCESS ;
- BEC T TERMINATES SALARIED-PAY-COMPUTATION, HOURLY-PAY-COMPUTATION ;
- BECG F TRMS ERROR-HANDLER ;

BECOMING TRIGGERS statement

CONDITION section

Purpose:

To specify a PROCESS/PROCESSES that are triggered by a change in state for this CONDITION.

Syntax:

```
BECOMING { TRUE }  
           { FALSE } TRIGGERS process-name(s);
```

Complementary Statements:

TRIGGERED statement in the PROCESS section.

Usage Rules:

- A CONDITION BECOMING TRUE or FALSE may TRIGGER several PROCESSES.
- A CONDITION BECOMING TRUE may TRIGGER one set of PROCESSES and BECOMING FALSE may TRIGGER a second set.

Synonyms:

```
{ BEC }  
{ } TRGS  
{ BECG }
```

Examples:

- BECOMING TRUE TRIGGERS BILLING-PROCESS ;
- BEC T TRIGGERS SALARIED-PAY-COMPUTATION, HOURLY-PAY-COMPUTATION ;
- BECG F TRGS ERROR-HANDLER ;

DESCRIPTION statement

CONDITION section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

Ncne.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

CONDITION section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a KEYWORD.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MADE statement

CONDITION section

Purpose:

To specify those EVENT(S), INPUT(S), and PROCESS(ES) which may set this CONDITION and to indicate the value to which it is set.

Syntax:

```
MADE { TRUE } event-  
      { FALSE } BY input-name(s) ;  
      process-
```

Complementary Statements:

MAKES statement in EVENT, INPUT, and PROCESS sections.

Usage Rules:

- A CONDITION may be set by several EVENTS.
- A CONDITION may be MADE TRUE by one set of EVENTS and MADE FALSE by another set of EVENTS.

Synonyms:

None.

Examples:

- MADE FALSE BY INPUT-ARRIVAL;
- MADE FALSE BY INPUT-ERROR, PROCESSING-ERROR;
- MADE TRUE BY ERROR-OCCURRENCE;

RESPONSIBLE-PROBLEM-DEFINER statement

CONDITION section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RFD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

CONDITION section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

CONDITION section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

-A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

CONDITION section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

CONDITION section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms (e.g. Abbreviations) for section names in the documentation. A synonym can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section .

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE C-11, CONDITION-11;
- SYNONYM IS CONDITION-11;
- SYN ALPHA;

TRACE-KEY statement

CONDITION section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name (s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

WHILE statement

CONDITION section

Purpose:

To give an expression on which this CONDITION depends.

Syntax:

```
{ TRUE }  
{ FALSE } WHILE ;  
comment-entry ;
```

Complementary Statements:

None.

Usage Rules:

-May be given only once for any CONDITION.

Synonyms:

```
{ T }  
{ } WHL  
{ F }
```

Examples::

- TRUE WHILE;
STILL AN EMPLOYEE;
- FALSE WHILE;
SYSTEM-BEING-UPDATED;
- T WHL;
SYSTEM OUTPUT STILL VALID;

4.2 DEFINE section header statement

Purpose:

To describe in greater detail certain name types within URL.
For example, if one wished to show a value or range of values for a system parameter, it would be done in this section.

Syntax:

	{ <u>ATTRIBUTE</u>	}[{ <u>ATTRIBUTE</u>	}]	
	{ <u>ATTRIBUTE-VALUE</u>	}[{ <u>ATTRIBUTE-VALUE</u>	}]	
	{ <u>CLASSIFICATION</u>	}[{ <u>CLASSIFICATION</u>	}]	
	{ <u>KEYWORD</u>	}[{ <u>KEYWORD</u>	}]	
	{ <u>MAILBOX</u>	}[{ <u>MAILBOX</u>	}]	
<u>DEFINE</u> name	{ <u>SECURITY</u>	}[, name	{ <u>SECURITY</u>	}]	... ;
	{ <u>SOURCE</u>	}[{ <u>SOURCE</u>	}]	
	{ <u>SUBSETTING-CRITERION</u>	}[{ <u>SUBSETTING-CRITERION</u>	}]	
	{ <u>SYSTEM-PARAMETER</u>	}[{ <u>SYSTEM-PARAMETER</u>	}]	
	{ <u>TRACE-KEY</u>	}[{ <u>TRACE-KEY</u>	}]	

Usage Rules:

- It must be the first statement in the DEFINE section.
- Several names may be defined at once.

Synonyms:

	{ ATTR	}
	{ ATTRV	}
	{ CLS CLASSIFICATIONS	}
	{ KEY	}
	{ BOX MBX	}
DEF	{ SEC	}
	{ SRC	}
	{ SSCN	}
	{ SYSP SYSPAR SYSTEM-PARAMETERS	}
	{ TKEY	}

Examples:

```
- DEFINE NAME-A ATTRIBUTE .....DEF NAME-A ATTR
- DEFINE NAME-B ATTRIBUTE-VALUE .....DEF NAME-B ATTRV
- DEFINE NAME-C CLASSIFICATION .....DEF NAME-C CLS
- DEFINE NAME-D KEYWORD .....DEF NAME-D KEY
- DEFINE NAME-E MAILBOX .....DEF NAME-E MBX
- DEFINE NAME-F SECURITY .....DEF NAME-F SEC
- DEFINE NAME-G SOURCE .....DEF NAME-G SRC
- DEFINE NAME-H SUBSETTING-CRITERION ..DEF NAME-H SSCN
- DEFINE NAME-I SYSTEM-PARAMETER .....DEF NAME-I SYSP
```

- DEFINE NAME-J TRACE-KEYDEF NAME-J TKEY

APPLIES statement

DEFINE section

Purpose:

To tie the information contained in the DEFINE section to any new or revised sections to which it applies.

Syntax:

APPLIES TO name(s) ;

Complementary Statements:

KEYWORDS, MAILBOX, SECURITY, SOURCE AND TRACE-KEY statements.

Usage Rules:

-This statement may only be given in the DEFINE sections for those names which are of the type KEYWORD, SECURITY, SOURCE, MAILBOX, or TRACE-KEY.

-The statement may be given as many times as necessary for the name.

-Multiple APPLIES statements for the same name are equivalent to a single statement with all the names in the list.

Synonyms:

APP

Examples:

-APPLIES TO NETWORK-IDENT;

-APPLIES TO NETWORK-IDENT, COMPANY-AND-AREA, TYPE-MATERIAL;

-APP PROCESS-1;

-APP TO NETWORK-IDENT, COMPANY-AND-AREA, TYPE-MATERIAL;

ASSEPT statement

DEFINE section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSEPT name attribute-name attribute-value

[, name attribute-name attribute-value];

Complementary Statements:

Ncne.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSEPT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

DEFINE section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ] ...  
                        { integer } [ { integer } ]
```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

DESCRIPTION statement

DEFINE section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

DEFINE section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a KEYWORD.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MAINTAINED statement

DEFINE section

Purpose:

To give the PROCESSES which maintain a SUBSETTING-CRITERION.

Syntax:

MAINTAINED BY process-name(s) ;

Complementary Statements:

MAINTAINS statement in PROCESS section.

Usage Rules:

-A SUBSETTING-CRITERION can be MAINTAINED by more than one PROCESS.

-THIS STATEMENT MAY ONLY BE USED TO DESCRIBE subsetting-criterion NAMES.

Synonyms:

MIND

Examples:

- MAINTAINED BY FIRST-PROCESS;

- MIND PROCESS-A, PROCESS-B;

RESPONSIBLE-PROBLEM-DEFINER statement

DEFINE section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RFD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RFD A-HERSHEY;

SECURITY statement

DEFINE section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

DEFINE section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

-A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

DEFINE section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SPC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBSETTING-CRITERION statement

DEFINE section

Purpose:

To indicate that this name is used to extract information from a SET to produce a SUBSET.

Syntax:

SUBSETTING-CRITERION FOR set-name(s) ;

Complementary Statements:

SUBSETTING-CRITERIA statement in a SET section.

Usage Rules:

-The names must be SET names.

-This statement may only be used to describe SUBSETTING-CRITERION names. -A name so defined may be a SUBSETTING-CRITERION for more than one SET.

Synonyms:

SSCN

Examples:

- SUBSETTING-CRITERION FOR SET-GROUP-BANKS, SET-GROUP-CKTS;
- SSCN: FILE-107, FILE-108;

SYNONYMS statement

DEFINE section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms (e.g. Abbreviations) for section names in the documentation. A synonym can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section .

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE ATTR-11, ATTRIBUTE-11;
- SYNONYM IS CLASSIFICATION-11;
- SYN ALPHA;

TRACE-KEY statement

DEFINE section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

VALUES statement

DEFINE section

Purpose:

To specify the allowable range of VALUES, or specific VALUES, which this SYSTEM-PARAMETER is free to take on. This is useful in determining the need to check data for validity within the system.

Syntax:

```

                {          integer          }
                {                          }
VALUES ARE { { min    }          { max  } } ;
                { {      } THRU   {      } }
                { { NEGINF }      { POSINF } }
```

Complementary Statements:

None.

Usage Rules:

- min and max must be integers
- Each min must be less than the corresponding max.

Synonyms:

VAL VALUE

Examples:

- VALUE 107;
- VALUES ARE 1 THRU 9999;
- VALUE NEGINF THRU POSINF;

4.3 DESIGNATE section header statement

Purpose:

To add additional SYNONYMS to names which already exist within the URL data base. This section is useful in standardizing system names, since one accepted name can be referred to by several different SYNONYM names.

Syntax:

```
DESIGNATE name AS A SYNONYM FOR name  
[ , name AS A SYNONYM FOR name ] ... ;
```

Usage Rules:

- No other statements are allowed in a DESIGNATE section.
- The first name in each pair is taken to be a synonym for the second name in the pair.

Synonyms:

```
DESG      SYN
```

Examples:

- DESIGNATE PROC-1 AS A SYNONYM FOR PROCESS-ONE;
- DESIGNATE A-1 AS A SYNONYM FOR ALPHA-MASTER;
- DESG R-1 SYN REPORT-FOR-NEW-MASTER-INPUT;

4.4 ELEMENT section header statement

Purpose:

To allow a detailed description of an ELEMENT. The element is the smallest item of data that can be referred to within the system and still maintain its unique properties.

Syntax:

ELEMENT element-name(s) ;

Usage Rules:

- Must be the first statement in an ELEMENT section.
- Several ELEMENTS may be defined at once.

Synonyms:

ELE ELEMENTS

Examples:

- ELEMENT CHECK-NUMBER;
- ELEMENTS SPAN-NUMBER, SPAN-MILEAGE;
- ELE EMPLOYEE-NUMBER;

ASSERT statement

ELEMENT section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

```
ASSERT name attribute-name attribute-value  
        [, name attribute-name attribute-value] ...;
```

Complementary Statements:

Ncne.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
 coord-function arguments 2;

ASSOCIATED statement

ELEMENT section

Purpose:

To show that the ELEMENT is jointly owned by two ENTITIES which have been described as having a relationship to each other through a RELATION section.

Syntax:

ASSOCIATED WITH relation-name(s) ;

Complementary Statements:

ASSOCIATED-DATA statement in the RELATION section.

Usage Rules:

- Name(s) must be RELATION names.
- An ELEMENT may be associated with several RELATIONS.

Synonyms:

ASOC

Examples:

- ASSOCIATED WITH RELATION-A;
- ASSOCIATED WITH NETWORK-RELATION, DERIVED-RELATION;
- ASOC RELATION-1, RELATION-2;
- ASOC NEW-RELATION;

ATTRIBUTES statement

ELEMENT section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```

ATTRIBUTES ARE attr-name { attrv-name } [
                        { integer } [ ,attr-name { attrv-name } ] ..
                        { integer } [

```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CLASSIFICATION statement

ELEMENT section

Purpose:

To associate security CLASSIFICATION requirements with data in the target system.

Syntax:

```
CLASSIFICATION classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

Ncne.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

CLS CLASSIFICATIONS

Examples:

- CLASSIFICATION IS PERSONNEL, SEC-LEVEL 3;
- CLS RING-LEVEL 2, UPDATE;

CONTAINED statement

ELEMENT section

Purpose:

To give the GROUPS, ENTITIES, INPUTS, and/or OUTPUTS that contain this ELEMENT. An ELEMENT being contained in a GROUP, ENTITY, INPUT, or OUTPUT means that the data values contained in the ELEMENT will be included in the logical GROUP, ENTITY, INPUT, or OUTPUT.

Syntax:

```
group-  
entity-  
CONTAINED IN input-name(s) ;  
output-
```

Complementary Statements:

CONSISTS statement in the GROUP, ENTITY, INPUT, and OUTPUT sections.

Usage Rules:

- The names must be GROUP, ENTITY, INPUT, or OUTPUT names.
- Several GROUPS, ENTITIES, INPUTS, or OUTPUTS may contain an ELEMENT.

Synonyms:

CNTD

Examples:

- CONTAINED IN GROUP-A1;
- CONTAINED IN ENTITY-1, ENTITY-2;
- CNTD IN INPUT-A;

UNCLASSIFIED

ESD-TR-78-129-VOL-2 NL

AD
AO6068-1

100

100

DERIVED statement

ELEMENT section

Purpose:

To give a PROCESS that DERIVES values for the ELEMENT and, optionally, the SETS, INPUTS, ENTITIES, GROUPS, and/or ELEMENTS used in the derivation.

Syntax:

```

DERIVED BY process-name(s) [
                                [ group-
                                [ entity-
                                [ USING set-name(s)
                                [ input-
                                [ element-
                                ]
                                ]
                                ]
                                ] ;

```

Complementary Statements:

DERIVES or USES statement in a PROCESS section and USED BY statement in a SET, INPUT, ENTITY, GROUP or ELEMENT section. - Several PROCESSES may derive an ELEMENT.

Synonyms:

DFVD USG

Examples:

- DERIVED BY PROCESS-A USING INPUT-1;
- DERIVED BY PROCESS-1 USING ENTITY-A, ENTITY-B;
- DFVD PROCESS-Q USG INPUT-1;
- DFVD PROCESS-NAME USG ENTITY-A, GROUP-B;

DESCRIPTION statement

ELEMENT section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

IDENTIFIES statement

ELEMENT section

Purpose:

To highlight the fact that this ELEMENT is being used within the system to identify data for storage, retrieval, or processing. This ELEMENT may be considered to be a key.

Syntax:

IDENTIFIES entity-name(s) ;

Complementary Statements:

IDENTIFIED statement in the ENTITY section.

Usage Rules:

- The names must be ENTITY names.
- An ELEMENT may be a potential IDENTIFIER for more than one ENTITY.

Synonyms:

IDS

Examples:

- IDENTIFIES ENT-47;
- IDENTIFIES ENT-784, ENT-6387;
- IDS ENT-957;

KEYWORDS statement

ELEMENT section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RESPONSIBLE-PROBLEM-DEFINER statement

ELEMENT section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- It may be used in any section except the PROBLEM-DEFINER section.
- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

FPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HEPSHEY;

SECURITY statement

ELEMENT section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

ELEMENT section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PPOJ-MGR-106, PPOJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

ELEMENT section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBSETTING-CRITERION statement

ELEMENT section

Purpose:

To indicate that this ELEMENT is used to extract information from a SET to produce a SUBSET.

Syntax:

SUBSETTING-CRITERION FOR set-name(s) ;

Complementary Statements:

SUBSETTING-CRITERIA statement in SET section.

Usage Rules:

- The names must be SET names.
- An ELEMENT may be a SUBSETTING-CRITERION for more than one SET.

Synonyms:

SSCN

Examples:

- SUBSETTING-CRITERION FOR SET-GROUP-BANKS, SET-GROUP-CKTS;
- SSCN: FILE-107, FILE-108;

SYNONYMS statement

ELEMENT section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE E-11, ELEMENT-11;
- SYNONYM IS ELEMENT-11;
- SYN ALPHA;

TRACE-KEY statement

ELEMENT section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

UPDATED statement

ELEMENT section

Purpose:

To indicate those PROCESSES which UPDATE this ELEMENT, and optionally, to specify the data used to do the updating.

Syntax:

```

        [      group-      ]
        [      entity-     ]
    UPDATED BY  process-name (s) [ USING element-  name (s) ] ;
        [      input-      ]
        [      set-        ]

```

Complementary Statements:

UPDATES or USES statement in PROCESS section and USED BY statement in INPUT, SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-An ELEMENT may be updated by more than one PROCESS.

Synonyms:

UPDD USG

Examples:

- UPDATED BY P-101;
- UPDD P-103, OUTPUT-P-675354 USING MASTER-FILE-6;

USED statement

ELEMENT section

Purpose:

To indicate the PROCESS(ES) that USE(D) this ELEMENT, and optionally, DERIVE(S) OUTPUTS or UPDATE(S) SETS, ENTITIES, GROUPS, and/or ELEMENTS.

Syntax:

	set-	[set-]
	input-	[{ <u>DERIVE</u> }	*output-]
<u>USES</u>	element-name(s)	[TO {		element-	name(s)] :
	group-	[{ <u>UPDATE</u> }	group-]
	entity-	[entity-]

* Output-name(s) may only be used with the DERIVE clause.

Complementary Statements:

USES, UPDATES or DERIVES statement in a PROCESS section and DERIVED or UPDATED statement in SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-Several PROCESSES may use the ELEMENT.

Synonyms:

DFV UPD

Examples:

- USED BY PROCESS-UPDATE;
- USED BY LINEAR-PROCESS, INTEGER-PROCESS TO DERIVE ALPHA;

VALUES statement

ELEMENT section

Purpose:

To specify the allowable range of VALUES, or specific VALUES, which this ELEMENT is free to take on. This is useful in determining the need to check data for validity within the system.

Syntax:

```

VALUES ARE { integer }
           { { min } { max } } ;
           { { } THRU { } }
           { { NEGINF } { POSINF } }

```

Complementary Statements:

Ncne.

Usage Rules:

- min and max must be integers
- Each min must be less than the corresponding max.

Synonyms:

VAL VALUE

Examples:

- VALUE 107;
- VALUES ARE 1 THRU 9999;
- VALUE NEGINF THRU POSINF;

4.5 ENTITY section header statement

Purpose:

To allow a detailed description of the contents of an ENTITY.
An ENTITY is a logical, usable collection of data that serves a unique purpose within the system. An ENTITY is information used by the target system that represents an object or concept of the real world. It is required by the target system for information processing purposes.

Syntax:

ENTITY entity-name(s) ;

Usage Rules:

- It must be the first statement in an ENTITY section.
- Several ENTITIES may be defined at once.

Synonyms:

ENT ENTITIES

Examples:

- ENTITY ROOT-SEGMENT;
- ENTITY NH-SEGMENT, NI-SEGMENT;
- ENT ENTITY-1;
- ENT NS-SEGMENT, NP-SEGMENT;

ASSERT statement

ENTITY section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value
[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

ENTITY section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attr-name } [ { attr-name } ]  
                                { attr-name } [ ,attr-name { attr-name } ] ..  
  
                                { integer } [ { integer } ]
```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CARDINALITY statement

ENTITY section

Purpose:

To define the number of times this ENTITY appears in the system. This can be used to estimate the size of SETS that contain the ENTITY.

Syntax:

CARDINALITY IS system-parameter ;

Complementary Statements:

None.

Usage Rules:

-An ENTITY may only have one CARDINALITY.

Synonyms:

CARD OCCS OCCURRENCES

Examples:

- CARDINALITY IS ONE;
- CARD ONE;

CLASSIFICATION statement

ENTITY section

Purpose:

To associate security CLASSIFICATION requirements with data in the target system.

Syntax:

```
CLASSIFICATION classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

CIS CLASSIFICATIONS

Examples:

- CLASSIFICATION IS PERSONNEL, SEC-LEVEL 3;
- CLS RING-LEVEL 2, UPDATE;

CONSISTS statement

ENTITY section

Purpose:

To describe the combination of GROUPS and/or ELEMENTS which make up this ENTITY. This implies that each instance of the ENTITY will contain values of the GROUP and ELEMENT names. A GROUP or ELEMENT may be repeated the number of times denoted by the SYSTEM-PARAMETER.

Syntax:

```
CONSISTS OF [ system-parameter ] element-
                                     group-name
                                     element-
[ , [ system-parameter ] group-name ] ... ;
```

Complementary Statements:

CONTAINED statement in the GROUP and ELEMENT sections.

Usage Rules:

- The names, other than the SYSTEM-PARAMETERS , must be GROUP or ELEMENT names.
- An ENTITY can contain several GROUPS or ELEMENTS.

Synonyms:

CSTS

Examples:

- CONSISTS OF ONE GR-1, ONE GR-2, TWO ELE-5 ;
- CONSISTS OF: UNIQUE-SPAN-NUMBER;
- CSTS TWO ELE-A, GROUP-7 ;

CONTAINED statement

ENTITY section

Purpose:

To give the SETS that contain this ENTITY. An ENTITY being contained in a SET means that the data values contained in the ENTITY will be included in the logical SET.

Syntax:

CCONTAINED IN set-name(s) ;

Complementary Statements:

CCONSISTS statement in a SET section.

Usage Rules:

- The names must be SET names.
- An ENTITY can be contained in several SETS.

Synonyms:

CNTD

Examples:

- CONTAINED IN INPUT-HS;
- CONTAINED IN: HS-1, HS-2, HS-3;
- CNTD IN FIRST-HS;
- CNTD: HS-ONE, OUTPUT-HS-ONE;
- CNTD: MASTER-FILE;
- CONTAINED PAYROLL-CHANGE, NAME-DELETE;
- CNTD NEW-EMPLOYEE;

DERIVED statement

ENTITY section

Purpose:

To give a PROCESS that DERIVES values for the ENTITY and, optionally, the SETS, INPUTS, ENTITIES, GROUPS, and/or ELEMENTS used in the derivation.

Syntax:

```

DERIVED BY process-name(s) [
    [ group-                ]
    [ entity-               ]
    [ USING set-name(s)    ] ;
    [ input-               ]
    [ element-             ]

```

Complementary Statements:

DERIVES or USES statement in a PROCESS section and USED BY statement in a SET, INPUT, ENTITY, GROUP or ELEMENT section.

Usage Rules:

-Several PROCESSES may derive an ENTITY.

Synonyms:

DEVD USG

Examples:

- DERIVED BY A-PROCESS USING ELE-1;
- DERIVED B-PROCESS USING ENTITY-456;
- DEVD OUT-PROCESS USG GROUP-SPAN-13;

DESCRIPTION statement

ENTITY section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

IDENTIFIED statement

ENTITY section

Purpose:

To give the possible GROUPS and/or ELEMENTS which identify this ENTITY. This is necessary to uniquely distinguish multiple instances of the same ENTITY. This statement can be viewed as defining a unique key for information retrieval purposes.

Syntax:

group
IDENTIFIED BY element-name(s) ;

Complementary Statements:

IDENTIFIES statements in GROUP and ELEMENT sections.

Usage Rules:

- The names must be either GROUP or ELEMENT names.
- An ENTITY may have several alternative identifiers.
- If the ENTITY is IDENTIFIED by a GROUP then the ELEMENTS which make up the GROUP are taken together as an identifier.

Synonyms:

IDD

Examples:

- IDENTIFIED BY SPAN-NUMBER;
- IDENTIFIED BY SPAN-NUMBER, SPAN-LOG;
- IDD ELEMENT-1, GROUP-1;

KEYWORDS statement

ENTITY section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RELATED statement

ENTITY section

Purpose:

To identify which RELATIONS and ENTITIES this ENTITY is associated with.

Syntax:

RELATED TO entity-name VIA relation-name ;

Complementary Statements:

BETWEEN statement in the RELATION section.

Usage Rules:

- The second name must be a RELATION name.
- The first name must be an ENTITY name.
- All RELATIONS are binary.

Synonyms:

FEL

Examples:

- RFLATED TO NH-ENTITY VIA UPDATE-RELATION;
- RFL NI-SEG VIA NI-RELATION;

RESPONSIBLE-PROBLEM-DEFINER statement

ENTITY section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

ENTITY section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

ENTITY section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

-SOURCES SDP-3-0;

-SEC ENG-LETTER-1-MAY-1973;

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

ENTITY section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

ENTITY section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- The statement may be used in any section except a MEMO section, or a DEFINE section for a SYNONYM.
- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE E-11, ENTITY-11;
- SYNONYM IS ENTITY-11;
- SYN ALPHA;

TRACE-KEY statement

ENTITY section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

UPDATED statement

ENTITY section

Purpose:

To indicate those PROCESSES which update this ENTITY, and optionally, to specify the data used to do the updating.

Syntax:

```
UPDATED BY process-name(s) [ [ group-          ]  
                             [ entity-         ]  
                             [ USING element-   name(s) ] ;  
                             [ input-          ]  
                             [ set-            ]
```

Complementary Statements:

UPDATES or USES statement in PROCESS section and USED BY statement in INPUT, SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-An ENTITY may be UPDATED by more than one PROCESS.

Synonyms:

UPDD USG

Examples:

- UPDATED BY P-101;
- UPDD P-103, OUTPUT-P-675354 USING MASTER-FILE-4;

USED statement

ENTITY section

Purpose:

To indicate the PROCESS(ES) that USE(D) this ENTITY, and optionally, DERIVE(S) OUTPUTS or UPDATE(S) SETS, ENTITIES, GROUPS,

Syntax:

```

                                [ set- ]
                                [ *output- ]
USED BY process-name(s) [ TO { DERIVE } entity- name(s) ] ;
                                [ { UPDATE } group- ]
                                [ element- ]

```

* Output-name(s) may only be used with the DERIVE clause.

Complementary Statements:

USES, UPDATES or DERIVES statement in a PROCESS section and DERIVED or UPDATED statement in SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-Several PROCESSES may use the ENTITY.

Synonyms:

DRV UPD

Examples:

- USED BY PROCESS;
- USED BY LINEAR-PROCESS, INTEGER-PROCESS TO UPDATE ENT-1;

VOLATILITY statement

ENTITY section

Purpose:

To give a measure of the changability of the ENTITY.

Syntax:

VOLATILITY ;
comment-entry ;

Complementary Statements:

Ncne.

Usage Rules:

-Only one VOLATILITY statement may be given for an ENTITY.

Synonyms:

VCL

Examples:

VOLATILITY;

SEGMENT IS UPDATED EACH TIME AN SR TRANSACTION IS REQUESTED;

4.6 EVENT section header statement

Purpose:

To describe the dynamic occurrences which take place within the target system. An EVENT is used to describe an instance of time during the operation of the target system. An EVENT may re-occur more than once during target system operation. For example, "occurrence of error" may be an EVENT which causes normal processing to be suspended while an error processor is initiated. An EVENT may occur when a PROCESS is started or finished, when a CONDITION becomes TRUE or FALSE, when an INPUT becomes available, or when another EVENT occurs.

Syntax:

EVENT event-name(s) ;

Usage Rules:

- It must be the first statement in an EVENT section.
- Several EVENTS may be defined at once.

Synonyms:

EV EVT EVFNTS

Examples:

- EVENT TIME-CARD-ENTRY;
- EVENTS REGISTER, CHECK-IN, CHECK-OUT;
- EV CARRIER-ALARM;
- EVT CARRIER-ALARM, CARRIER-FAILURE;

ASSERT statement

EVENT section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

EVENT section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ { attrv-name } ]  
                        { integer } [ ,attr-name { integer } ] ...
```

Complementary Statements:
none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAP ZZZ9V9;

CAUSED statement

EVENT section

Purpose:

To specify INPUT (S), CONDITION (S), or additional EVENT (S) which cause this EVENT.

Syntax:

```
event-  
CAUSED BY      name(s) ;  
input-
```

```
CAUSED WHEN condition-name BECOMES { TRUE }  
                                     { FALSE };
```

Complementary Statements:

CAUSES statement in the EVENT and INPUT sections, and BECOMING CAUSES statement in the CONDITION section.

Usage Rules:

- AN EVENT may be CAUSED by any number of EVENTS and/or INPUTS.
- A separate statement is required for each CONDITION change which CAUSES an EVENT. Any number of such statements may appear in a single EVENT section.

Synonyms:

CSD

Examples:

- CAUSED BY TIME-CARD-INPUT, DEADLINE-REACHED;
- CAUSED WHEN ERROR-FLAG-SET BECOMES TRUE;
- CSD ORDERS;

CAUSES statement

EVENT section

Purpose:

To specify other EVENT(S) which are caused by this EVENT.

Syntax:

CAUSES event-name (S) ;

Complementary Statements:

CAUSED statement in the EVENT section.

Usage Rules:

- An EVENT may CAUSE several other EVENTS.

Synonyms:

CSS

Examples:

- CAUSES SUBPROCESS-COMPLETION, MAIN-PROCESS-COMPLETION ;
- CSS ERROR-DETECTED ;

DESCRIPTION statement

EVENT section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

Ncne.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

HAPPENS statement

EVENT section

PURPOSE:

To give the number of times an EVENT occurs during an INTERVAL. More than one instance of an EVENT may occur over some period of time. The number of instances of the EVENT which occur in a time INTERVAL is expressed with this statement.

Syntax:

HAPPENS system-parameter TIMES-PER interval-name ;

Complementary Statements:

None.

Usage Rules:

- The name must be an INTERVAL name.
- The statement may be given as many times as necessary for different INTERVALS.

Synonyms:

HAP TIMP

Example:

- HAPPENS FORTY-SEVEN TIMES-PER INTERVAL-A;
- HAP THIRTY-TWO TIMP INT-B;

INCEPTION statement

EVENT section

Purpose:

To specify those PROCESS(ES) whose inception causes this EVENT.

Syntax:

ON INCEPTION OF process-name(s) ;

Complementary Statements:

INCEPTION-CAUSES statement in a PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- Several PROCESSES may be given.

Synonyms:

INCP

Examples:

- ON INCEPTION OF PROCESS-IN;
- INCEPTION OF PROCESS-OUT;
- INCP SORT-ALPHA;

INTERRUPTS statement

EVENT section

Purpose:

To specify those PROCESS(ES) which are interrupted as a result of this EVENT.

Syntax:

INTERRUPTS process-name(s);

Complementary Statements:

INTERRUPTED statement in the PROCESS section.

Usage Rules:

- An EVENT may INTERRUPT several PROCESSES.

Synonyms:

INTS

Examples:

- INTERRUPTS MAIN-PROCESSING ;
- INTS MASTER-FILE-SEARCH, PAYSYSTEM-PROCESSING;

KEYWORDS statement

EVENT section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a KEYWORD.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MAKES statement**EVENT section****Purpose:**

To give those **CONDITION(S)** which are set by this **EVENT**.

Syntax:

```
MAKES condition-name(s) { TRUE }  
                        { FALSE } ;
```

Complementary Statements:

MADE statement in the **CONDITION** section.

Usage Rules:

- An **EVENT** may make several **CONDITIONS** become **TRUE** or **FALSE**.
- An **EVENT** cannot **MAKE** some **CONDITION(S)** **TRUE** and other **CONDITION(S)** **FALSE** in the same statement. Separate statements are required.

Synonyms:

MAK

Examples:

- **MAKES PROCESS-COMPLETION TRUE ;**
- **MAK ERROR-OCCURRENCE, OUTPUT-INTERRUPTION F ;**

RESPONSIBLE-PROBLEM-DEFINER statement

EVENT section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RED

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

EVENT section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information , not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

EVENT section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

EVENT section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SPC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

EVENT section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE E-11, EVENT-11;
- SYNONYM IS EVENT-11;
- SYN ALPHA;

TERMINATES statement

EVENT section

Purpose:

To specify a PROCESS/PROCESSES that are terminated by this EVENT.

Syntax:

TERMINATES process-name(s) ;

Complementary Statements:

TERMINATED statement in PROCESS section.

Usage Rules:

- An EVENT may TERMINATE several PROCESSES.

Synonyms:

TRMS

Examples:

- TERMINATES INPUT-PROCESSING;
- TRMS PROC-A, PROC-B, PROC-C ;

TERMINATION statement

EVENT section

Purpose:

To indicate those PROCESS(ES) on whose TERMINATION this EVENT occurs.

Syntax:

ON TERMINATION OF process-name(s) ;

Complementary Statements:

TERMINATION-CAUSES statement in a PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- Several PROCESSES may be given.

Synonyms:

TERM

Examples:

- ON TERMINATION OF INPUT-PROCESS;
- TERMINATION UPDATE-PROCESS;
- TERM FORECAST-PROCESS;

TRACE-KEY statement

EVENT section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

TRIGGERS statement

EVENT section

Purpose:

To give the PROCESS/PROCESSES which are triggered when this EVENT occurs.

Syntax:

TRIGGERS process-name(s) ;

Complementary Statements:

TRIGGERED statement in PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- Several PROCESSES may be triggered by any EVENT.

Synonyms:

TRGS

examples:

- TRIGGERS UPDATE-PROCESS;
- TRIGGERS P-101, P-420, P-7598;
- TRGS EXTRA-LINK-PROCESS;

4.7 GROUP section header statement

Purpose:

To allow a detailed description of a GROUP. A GROUP is a logical collection of data ELEMENTS and/or other GROUPS. A GROUP is a collection of information which can be CONTAINED in larger collections of information. E.g. INPUTS, OUTPUTS, and ENTITIES. For instance, current-date might be a GROUP containing month, day and year.

Syntax:

GROUP group-name (s) ;

Usage Rules:

- It must be the first statement in a GROUP section.
- Several GROUPS may be defined at once.

Synonyms:

GR GROUPS

Examples:

- GROUP SPAN-MAKEUP;
- GROUPS: SPAN-A, LINK-A;
- GR GROUP-A;
- GR: SPAN-784, LINK-737;

ASSERT statement

GROUP section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

```
ASSERT name attribute-name attribute-value  
        [, name attribute-name attribute-value] ...;
```

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
 coord-function arguments 2;

ASSOCIATED statement

GROUP section

Purpose:

To show that the GROUP is jointly owned by two ENTITIES which have been described as having a relationship to each other through a RELATION section.

Syntax:

ASSOCIATED WITH relation-name(s) ;

Complementary Statements:

ASSOCIATED-DATA statement in a RELATION section.

Usage Rules:

- The names must be RELATION names.
- A GROUP may be associated with several RELATIONS.

Synonyms:

ASOC

Examples:

- ASSOCIATED WITH EMPLOYED-BY-RELATION;
- ASSOCIATED WITH NAME-RELATION, DATE-RELATION, TIME-RELATION;
- ASOC RELATION-C1;
- ASOC RELATION-C1,RELATION-C2,RELATION-C3;

ATTRIBUTES statement

GROUP section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attr-name } [
{ integer } [ ,attr-name { attr-name } ] ] ..
{ integer } ] ]
```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CLASSIFICATION statement

GROUP section

Purpose:

To associate security CLASSIFICATION requirements with data in the target system.

Syntax:

```
CLASSIFICATION classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

CLS CLASSIFICATIONS

Examples:

- CLASSIFICATION IS PERSONNEL, SEC-LEVEL 3;
- CLS RING-LEVEL 2, UPDATE;

CONSISTS statement

GROUP section

Purpose:

To describe the combination of other GROUPS and/or ELEMENTS which make up this GROUP. This implies that each instance of the GROUP will contain values of the GROUP and ELEMENT names. A GROUP or ELEMENT may be repeated the number of times denoted by the SYSTEM-PARAMETER.

Syntax:

```
                                element-  
CONSISTS OF [ system-parameter ] group-name  
                                element-  
                                [ , [ system-parameter ] group-name ] ... ;
```

Complementary Statements:

CONTAINED statement in a GROUP or ENTITY section.

Usage Rules:

- The names, other than the system-parameters, must be GROUP or ELEMENT names.
- A GROUP can contain several GROUPS or ELEMENTS.

Synonyms:

CSTS

Examples:

- CONSISTS OF TWO DATA-GROUP-1;
- CONSISTS: DATA-GROUP-1, ELEMENT-A;
- CSTS OF SPAN-ELEMENT-A;
- CSTS: GROUP-NO-1, GROUP-NO-2;

CONTAINED statement

GROUP section

Purpose:

To give the ENTITIES, INPUTS, OUTPUTS, or GROUPS that contain this GROUP. A GROUP being contained in a GROUP, ENTITY, INPUT, or OUTPUT means that the data values contained in the GROUP will be included in the logical GROUP, ENTITY, INPUT, or OUTPUT.

Syntax:

```
group-  
entity-  
CONTAINED IN    -name(s) ;  
input-  
output-
```

Complementary Statements:

CONSISTS statement in GROUP, ENTITY, INPUT and OUTPUT sections .

Usage Rules:

- The names must be GROUP, ENTITY, INPUT or OUTPUT names.
- A GROUP may be contained in several GROUPS, ENTITIES, INPUTS or OUTPUTS.

Synonyms:

CNTD

Examples:

- CONTAINED IN GROUP-1;
- CONTAINED IN GROUP-2, INPUT-2, OUTPUT-REP;
- CNTD IN FIRST-ENTITY;

DERIVED statement

GROUP section

Purpose:

To give a PROCESS that DERIVES values for the GROUP and, optionally, the SETS, INPUTS, ENTITIES, GROUPS, and/or ELEMENTS used in the derivation.

Syntax:

```
DERIVED BY process-name(s) [ [ group-          ]  
                             [ entity-         ]  
                             [ USING set-name(s) ] ;  
                             [ input-          ]  
                             [ element-        ]
```

Complementary Statements:

DERIVES or USES statement in a PROCESS section and USED BY statement in a SET, INPUT, ENTITY, GROUP or ELEMENT section.

Usage Rules:

-Several PROCESSES may derive a GROUP.

Synonyms:

DRVD USG

Examples:

- DERIVED BY PROC-NAME USING GROUP-22;
- DERIVED BY PAYROLL-PROCESSING USING PAY-MAST, PAY-STMT;
- DRVD SPAN-UPDATE USG SPAN-NO, MILES;

DESCRIPTION statement

GROUP section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

IDENTIFIES statement

GROUP section

Purpose:

To highlight the fact that this GROUP is being used within the system to identify data for storage, retrieval, or processing. This GROUP may be considered to be a key in the target system.

Syntax:

IDENTIFIES entity-name(s) ;

Complementary Statements:

IDENTIFIED statement in ENTITY section.

Usage Rules:

- The names must be ENTITY names.
- A GROUP may IDENTIFY several different ENTITIES.
- If an ENTITY is identified by a GROUP, then the ELEMENTS which make up the GROUP taken together form the identifier.

Synonyms:

IDS

Examples:

- IDENTIFIES ENTITY-743;
- IDENTIFIES ENTITY-78954, ENTITY-8;
- IDS ENT-3;

KEYWORDS statement

GROUP section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RESPONSIBLE-PROBLEM-DEFINER statement

GROUP section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

GROUP section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

GROUP section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

GROUP section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBSETTING-CRITERION statement

GROUP section

Purpose:

To indicate that this GROUP is used to extract information from a SET to produce a SUBSET.

Syntax:

SUBSETTING-CRITERION FOR set-name(s) ;

Complementary Statements:

SUBSETTING-CRITERIA statement in SET section.

Usage Rules:

- The names must be SET names.
- A GROUP may be a SUBSETTING-CRITERION for more than one SET.
- If a GROUP is a SUBSETTING-CRITERION then the ELEMENTS which make up the GROUP taken together form the SUBSETTING-CRITERION for that SET.

Synonyms:

SSCN

Examples:

- SUBSETTING-CRITERION FOR HS-GROUP-BANKS, HS-GROUP-CKTS;
- SSCN: HS-GROUP-107, HS-GROUP-108;

SYNONYMS statement

GROUP section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE G-11, GROUP-11;
- SYNONYM IS GROUP-11;
- SYN ALPHA;

TRACE-KEY statement

GROUP section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

UPDATED statement

GROUP section

Purpose:

To indicate those PROCESSES which update this GROUP, and optionally, to specify the data used to do the updating.

Syntax:

```
UPDATED BY process-name(s) [ [ group-          ]  
                             [ entity-         ]  
                             [ USING element-   name(s) ] ;  
                             [ input-          ]  
                             [ set-            ]
```

Complementary Statements:

UPDATES or USES statement in PROCESS section and USED BY statement in INPUT, SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-A GROUP may be UPDATED by more than one PROCESS.

Synonyms:

UPDD USG

Examples:

- UPDATED BY P-101;
- UPDD P-103, OUTPUT-P-675354 USING FILE-A;

USED statement

GROUP section

Purpose:

To indicate the PROCESS(ES) that USE(D) this GROUP, and optionally, DERIVE(S) OUTPUTS or UPDATE(S) SETS, ENTITIES, GROUPS,

Syntax:

```

                                [ set-
USED BY process-name(s) [ TO [ { DERIVE } *output-
                                [ { UPDATE } entity- name(s) ] ;
                                [ group-
                                [ element-

```

* Output-name(s) may only be used with the DERIVE clause.

Complementary Statements:

USES, UPDATES or DERIVES statement in a PROCESS section and DERIVED or UPDATED statement in SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-Several PROCESSES may use the GROUP.

Synonyms:

DFV UPD

Examples:

- USED BY PROCESS-A;
- USED BY LINEAR-PROCESS, INTEGER-PROCESS TO UPDATE GR-4;

4.8 INPUT section header statement

Purpose:

To allow a detailed description of an INPUT. An INPUT is used to describe a collection of information produced external to the target system but used by the target system. An INPUT shows the flow of data from the outside world into the system. Hence, it crosses the system boundary. The INPUT section is also used to uniquely identify each system input.

Syntax:

INPUT input-name(s) ;

Usage Rules:

- Must be the first statement in a INPUT section.
- Several INPUTS may be defined at a time.

Synonyms:

INP

Examples:

- INPUT PAYROLL-CODE;
- INPUT CODE ;
- INP DATA-FOR-COMMUNICATION;

ASSERT statement

INPUT section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSEPT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSEPT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

INPUT section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attr-name } [ { attr-name } ]  
                        { integer } [ ,attr-name { attr-name } ] ..  
                        { integer } [ { integer } ]
```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CAUSES statement

INPUT section

Purpose:

To specify an EVENT/EVENTS which are caused by this INPUT.

Syntax:

CAUSES event-name(s) ;

Complementary Statements:

CAUSED statement in the EVENT section.

Usage Rules:

- An INPUT may CAUSE several EVENTS.

Synonyms:

CSS

Examples:

- CAUSES START-PROC-A;
- CSS SUBPROCESS-COMPLETION, MAIN-PROCESS-BEGINS ;

CLASSIFICATION statement

INPUT section

Purpose:

To associate security CLASSIFICATION requirements with data in the target system.

Syntax:

```
CLASSIFICATION classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

CLS CLASSIFICATIONS

Examples:

- CLASSIFICATION IS PERSONNEL, SEC-LEVEL 3;
- CLS RING-LEVEL 2, UPDATE;

CONSISTS statement

INPUT section

Purpose:

To describe the combination of GROUPS, and/or ELEMENTS which make up this INPUT. This implies that each instance of the INPUT will contain values of the GROUP and ELEMENT names. A GROUP or ELEMENT may be repeated the number of times denoted by the SYSTEM-PARAMETER.

Syntax:

```
          element-  
CONSISTS OF [ system-parameter ] group-name  
          element-  
          [ , [ system-parameter ] group-name ] ... ;
```

Complementary Statements:

CONTAINED statement in a GROUP or ELEMENT section.

Usage Rules:

- The names, other than the system-parameters, must be GROUP or ELEMENT names.
- An INPUT can contain several GROUPS or ELEMENTS.

Synonyms:

CSTS

Examples:

- CONSISTS OF TWO DATA-GROUP-1;
- CONSISTS: DATA-GROUP-1, ELEMENT-A;
- CSTS OF SPAN-ELEMENT-A;
- CSTS: GROUP-NO-1, GROUP-NO-2;

CONTAINED statement

INPUT section

Purpose:

To give the SETS that contain this INPUT. An INPUT being contained in a SET means that the data values contained in the INPUT will be included in the logical SET.

Syntax:

CONTAINED IN set-name(s) ;

Complementary Statements:

CCNSISTS statement in an SET section.

Usage Rules:

- The names must be SET names.
- Several SETS may contain a given INPUT.

Synonyms:

CNTD

Examples:

- CONTAINED IN MASTER-FILE;
- CNTD: HS-1,HS-2;
- CNTD FILE-1;

DESCRIPTION statement

INPUT section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

Ncne.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

GENERATED statement

INPUT section

Purpose:

To identify the INTERFACE which produces this INPUT for the system.

Syntax:

GENERATED BY interface-name(s) ;

Complementary Statements:

GENERATES statement in INTERFACE section.

Usage Rules:

-The names must be INTERFACE names.

Synonyms:

GEND

Examples:

- GENERATED BY INPUT-INTERFACE-1;
- GEND BY INTEFFACE-456;

HAPPENS statement

INPUT section

Purpose:

To give the volume of this INPUT. More than one instance of an INPUT may occur over some period of time. The number of instances of the INPUT which occur in a time INTERVAL is expressed with this statement.

Syntax:

HAPPENS system-parameter TIMES-PER interval-name ;

Complementary Statements:

Ncne.

Usage Rules:

- The name must be an INTERVAL name.
- The statement may be given as many times as necessary with different INTERVAL names.

Synonyms:

HAP TIMP

Examples:

- HAPPENS FORTY-SEVEN TIMES-PER INTERVAL-A;
- HAP THIRTY-TWO TIMP INT-B;

INTERUPTS statement

INPUT section

Purpose:

To specify those PROCESS (ES) which are interrupted by the arrival of this INPUT.

Syntax:

INTERUPTS process-name(s);

Complementary Statements:

INTERRUPTED statement in the PROCESS section.

Usage Rules:

- An INPUT may INTERRUPT several PROCESSES.

Synonyms:

INTS

Examples:

- INTERUPTS PAYCHECK-PROCESSING;
- INTS LOADING-PROC-A, LOADING-PROC-B, LOADING-PROC-C;

KEYWORDS statement

INPUT section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

- A section may have several KEYWORDS.

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MAKES statement

INPUT section

Purpose:

To give those CONDITION(S) which are set when this INPUT arrives.

Syntax:

```
MAKES condition-name(s) { TRUE }  
                        { FALSE } ;
```

Complementary Statements:

MADE statement in the CONDITION section.

Usage Rules:

- An INPUT may make several CONDITIONS become TRUE or FALSE.
- An INPUT cannot MAKE some CONDITION(S) TRUE and some CCNDITION(S) FALSE in a single statement. Separate statements are required.

Synonyms:

MAK

Examples:

- MAKES END-OF-FILE-REACHED, INPUT-PROC-COMPLETION TRUE ;
- MAK SYSTEM-READY FALSE;
- MAK FATAL-ERROR, PROGRAM-INTERRUPT T;

PART statement

INPUT section

Purpose:

To show the structural relationship of this INPUT to a higher-level INPUT. This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

PART OF input-name ;

Complementary Statements:

SUBPARTS statement in an INPUT section.

Usage Rules:

-The name must be an INPUT name.

-Only one INPUT name may be given, hence, only a tree structure can be established.

Synonyms:

none.

Examples:

- PART OF IN-101;

- PART INPUT-35;

RECEIVED statement

INPUT section

Purpose:

To show which PROCESS uses or receives the INPUT.

Syntax:

RECEIVED BY process-name(s) ;

Complementary Statements:

RECEIVES statement in PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- An INPUT may be received by more than one PROCESS.

Synonyms:

RCVD

Examples:

- RECEIVED BY P-104;
- RCVD P-89;

RESPONSIBLE-PROBLEM-DEFINER statement

INPUT section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- It may be used in any section except the PROBLEM-DEFINER section.
- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

INPUT section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SFC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SFC L-HANNON;

SEE-MEMO statement

INPUT section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

INPUT section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- It may be used in any section except a DEFINE section for a SOURCE.
- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBPARTS statement

INPUT section

Purpose:

To show the structural relationship of this INPUT to lower-level INPUT(S). This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBPARTS ARE input-name(s) ;

Complementary Statements:

PART statement in an INPUT section.

Usage Rules:

- The names must be INPUT names.
- An INPUT may be composed of several other INPUTS.

Synonyms:

SUBP

Examples:

- SUBPARTS ARE IN-101, IN-103;
- SUBP IN-309, INPUT-6785;

SYNONYMS statement

INPUT section

Purpose:

To give SYNONYMS for the name of the section. Can be used to defined short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE I-11, INPUT-11;
- SYNONYM IS INPUT-11;
- SYN ALPHA;

AD-A060 684 MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 F19628-76-C-0197

MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 F19628-76-C-0197

ESD-TR-78-129-VOL-2

NL

AD
AO6068-2

TERMINATES statement

INPUT section

Purpose:

To specify a PROCESS/PROCESSES that are terminated by this INPUT.

Syntax:

TERMINATES process-name(s) ;

Complementary Statements:

TERMINATED statement in PROCESS section.

Usage Rules:

- An INPUT may TERMINATE several PROCESSES.

Synonyms:

TFMS

Examples:

- TERMINATES PAYROLL-PROCESSING;
- TFMS PRINTING-PROCESS, PACKING-PROCESS;

TRACE-KEY statement

INPUT section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

TRIGGERS statement

INPUT section

Purpose:

To specify a PROCESS/PROCESSES that are triggered by this INPUT.

Syntax:

TRIGGERS process-name(s) ;

Complementary Statements:

TRIGGERED statement in the PROCESS section.

Usage Rules:

- An INPUT may TRIGGER several PROCESSES.

Synonyms:

TRGS

Examples:

- TRIGGERS MISSILE-CORRECTION, EVASIVE-MANEUVERS;
- TRGS MAIN-PROCESSING;

USED statement

INPUT section

Purpose:

To indicate the PROCESS(ES) that USE(D) this INPUT, and optionally, DERIVE(S) OUTPUTS or UPDATE(S) SETS, ENTITIES, GROUPS, or ELEMENTS.

Syntax:

```

                                [ set- ]
                                [ *output- ]
USED BY process-name(s) [ TO { DERIVE } entity- name(s) ] ;
                                [ { UPDATE } group- ]
                                [ element- ]

```

* Output-name(s) may only be used with the DERIVE clause.

Complementary Statements:

USES, UPDATES or DERIVES statement in a PROCESS section and DERIVED or UPDATED statement in SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

- Several PROCESSES may use the INPUT.

Synonyms:

DEV UPD

Examples:

- USED BY PROCESS;
- USED BY LINEAR-PROCESS, INTEGER-PROCESS TO DERIVE ALPHA;

4.9 INTERFACE section header statement

Purpose:

To allow a detailed description of an INTERFACE. The INTERFACE is an object, organization or system outside the boundaries of the target system that interacts with the system being described. It identifies the origin and destination of system products so that a complete understanding of the system may be obtained.

Syntax:

INTERFACE interface-name(s) ;

Usage Rules:

- Must be the first statement of every INTERFACE section.
- Several INTERFACES may be defined at once.

Synonyms:

INTF
INTERFACES
PWE
PEAL-WORLD-ENTITY
ORGU
ORGANIZATIONAL-UNIT

Examples:

- INTERFACE RWE-22;
- RWE PAYROLL;
- ORGANIZATIONAL-UNIT STENO-POOL;
- ORGU WAREHOUSE-4;

ASSERT statement

INTERFACE section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

```
ASSERT name attribute-name attribute-value  
      [, name attribute-name attribute-value] ...;
```

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASPT

Examples:

- ASSERT data-name-1 type character;
- ASSERT sine-function arguments 1,
 coord-function arguments 2;

ATTRIBUTES statement

INTERFACE section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attv-name } [
                                { integer } [ ,attr-name { attv-name } ]
                                { integer } [ { integer } ] ] ;
```

Complementary Statements:
none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

DESCRIPTION statement

INTERFACE section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION :
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;
THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT
THIS SECTION TO DO;

DESC;
ANY RELEVANT INFORMATION GOES HERE;

GENERATES statement

INTERFACE section

Purpose:

To give those INPUTS generated by this INTERPACE.

Syntax:

GENERATES input-name(s) ;

Complementary Statements:

GENERATED statement in INPUT section.

Usage Rules:

- The names must be INPUT names.
- A INTERFACE may generate several INPUTS.

Synonyms:

GENS

Examples:

- GENERATES SYSTEM-IN-1;
- GENERATES IN-A, IN-B;
- GENS SYSTEM-INPUT;
- GENS SYS-A-IN, SYS-B-IN;

KEYWORDS statement

INTERFACE section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

key keyword

Examples:

- keyword is payroll;
- key is con-cl;
- keywords are emp, empl, employee;

PART statement

INTERFACE section

Purpose:

To show the structural relationship of this INTERFACE to a higher-level INTERFACE. This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

PART OF interface-name ;

Complementary Statements:

SUBPARTS statement in an INTERFACE section.

Usage Rules:

- The name must be an INTERFACE name.
- Only one INTERFACE name can be given, hence, only a tree structure may be established.

Synonyms:

none.

Examples:

- PART OF PAYROLL-SYSTEM;
- PART DEPT-601;

RECEIVES statement

INTERFACE section

Purpose:

To identify the OUTPUTS produced by the system and show where they are used outside the system. This is necessary for a complete system definition.

Syntax:

RECEIVES output-name(s) ;

Complementary Statements:

RECEIVED BY statement in OUTPUT section.

Usage Rules:

- The names must be OUTPUT names.
- An INTERFACE may receive several OUTPUTS.

Synonyms:

RCVS

Examples:

- RECEIVES FORECAST-FILE-OUTPUT;
- RECEIVES OUTPUT-FILE-A, OUTPUT-FILE-B;
- RCVS OUT-1001, OUT-103;

RESPONSIBLE statement

INTERFACE section

Purpose:

To identify those SETS which this INTERFACE controls, maintains, and/or administers.

Syntax:

RESPONSIBLE FOR set-name(s) ;

Complementary Statements:

RESPONSIBLE-INTERFACE statement in SET section.

Usage Rules:

- The names must be SET names.
- An INTERFACE may be RESPONSIBLE for several SETS.

Synonyms:

RESP RES

Examples:

- RESPONSIBLE FOR PAYROLL-FILE;
- RESP FILE-A, FILE-B;

RESPONSIBLE-PROBLEM-DEFINER statement

INTERFACE section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

INTERFACE section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SECURITY-ACCESS-RIGHT statement

INTERFACE section

Purpose:

To give the type and level of security associated with an INTERFACE during operation of the target system.

Syntax:

```
SECURITY-ACCESS-RIGHT classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

SAR SECURITY-ACCESS-RIGHTS

Examples:

- SECURITY-ACCESS-RIGHTS ARE PERSONNEL, SEC-LEVEL 3;
- SAR RING-LEVEL 2, UPDATE;

SEE-MEMO statement

INTERFACE section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEP-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

INTERFACE section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBPARTS statement

INTERFACE section

Purpose:

To show the structural relationship of this INTERFACE to lower-level INTERFACE(S). This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBPARTS ARE interface-name(s) ;

Complementary Statements:

PART statement in an INTERFACE section.

Usage Rules:

- The names must be INTERFACE names.
- An INTERFACE may be composed of several other INTERFACES.

Synonyms:

SUBP

Examples:

- SUBPARTS ARE RWE-1, RWE-2;
- SUBP : PAYROLL-SYSTEM;

SYNONYMS statement

INTERFACE section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE I-11, INTERFACE-11;
- SYNONYM IS INTERFACE-11;
- SYN ALPHA;

TRACE-KEY statement

INTERFACE section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.10 INTERVAL section header statement

Purpose:

To allow a detailed description of an INTERVAL or INTERVALS. An INTERVAL is a specific duration of time or a time unit within the system. In defining frequency of an occurrence in the system, the frequency must be defined with respect to some time unit. For example, the designer might specify that a fiscal year lasted from June to May, and a calendar year from January to December.

Syntax:

INTERVAL interval-name(s) ;

Usage Rules:

- It must be the first statement in an INTERVAL section.
- Several INTERVALS may be defined at once.

Synonyms:

INT INTERVALS

Examples:

- INTERVAL WORK-WEEK;
- INTERVALS: BUSINESS-DAY, DAY;
- INT PERIOD-1;

ASSERT statement

INTERVAL section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

INTERVAL section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attr-name } [
                                { integer } [ ,attr-name { attr-name } ] ...
                                { integer } ]
```

Complementary Statements:

none.

Usage Rules:

- It may be used in any section.
- A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CONSISTS statement

INTERVAL section

Purpose:

To describe the combination of other^s INTERVALS which make up this INTERVAL. This implies that each instance of the INTERVAL will contain values of other INTERVAL names. An INTERVAL may be repeated the number of times denoted by the SYSTEM-PARAMETER.

Syntax:

CONSISTS OF [system-parameter] interval-name
[, [system-parameter] interval-name] ... ;

Complementary Statements:

None.

Usage Rules:

- The names, other than the SYSTEM-PARAMETERS , must be INTERVAL names.
- An INPUT may contain several INTERVALS.

Synonyms:

CSTS

Examples:

- CONSISTS OF INTERVAL-A;
- CONSISTS OF INTERVAL-1, INTERVAL-2;
- CSTS: SIXTY SECONDS, ONE HOUR;

DESCRIPTION statement

INTERVAL section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

INTERVAL section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RESPONSIBLE-PROBLEM-DEFINER statement

INTERVAL section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- It may be used in any section except the PROBLEM-DEFINER section.
- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

INTERVAL section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

INTERVAL section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

INTERVAL section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

INTERVAL section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- The statement may be used in any section except a MEMO section, or a DEFINE section for a SYNONYM.
- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE I-11, INTERVAL-11;
- SYNONYM IS INTERVAL-11;
- SYN ALPHA;

TRACE-KEY statement

INTERVAL section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.11 MEMO section header statement

Purpose:

To define MEMOS. A MEMO is a description relevant to one or more other objects in the target system. MEMOS can be used to record as part of the system documentation significant information which needs to be highlighted. This might include assumptions made during design, limitations assumed or known to exist (e.g. Hardware. They can also be used to record outstanding problems, requests, effective dates, etc.)

Syntax:

MEMO memo-name(s) ;

Usage Rules:

- It must be the first statement in a MEMO section.
- Several MEMOS may be defined at once.

Synonyms:

mcne.

Examples:

- MEMO NOTE-ON-UNRESOLVED-PROCESS-63;
- MEMO M-73, M-86;

APPLIES statement

MEMO section

Purpose:

To tie this MEMO to one or more sections so that a cross-reference to the MEMO appears in the documentation.

Syntax:

APPLIES TO non-memo-name(s) ;

Complementary Statements:

SEE-MEMO statement in all sections except the MEMO section.

Usage Rules:

-The names may be any type of name except a MEMO name.

Synonyms:

AER_q

Examples:

- APPLIES TO PROCESS-1, PROCESS-2;
- APPLIES TO FREQUENCY-BAND, PRICING-UNIT-NAME;
- APP NETWORK-SOURCE;
- APP LINK-IDENT, NETWORK-NOTES, BASE-NETWORK;

ASSERT statement

MEMO section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value
[, name attribute-name attribute-value] ...;

Complementary Statements:

Ncne.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

MEMO section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ]  
                        { integer } [ { integer } ] .
```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

DESCRIPTION statement

MEMO section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

MEMO section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RESPONSIBLE-PROBLEM-DEFINER statement

MEMO section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- It may be used in any section except the PROBLEM-DEFINER section.
- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

MEMO section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SOURCE statement

MEMO section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name (s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

MEMO section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- The statement may be used in any section except a DEFINE section for a SYNONYM.
- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE M-11, MEMO-11;
- SYNONYM IS MEMO-11;
- SYN ALPHA;

TRACE-KEY statement

MEMO section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.12 OUTPUT section header statement

Purpose:

To allow a detailed description of an OUTPUT. An OUTPUT is used to describe a collection of information produced by the target system, but is used external to that system. The OUTPUT section is used to show the flow of data from the system to the outside world. Hence, it crosses the system boundary. It can also be used to locate and uniquely identify each system output.

Syntax:

OUTPUT output-name(s) ;

Usage Rules:

-Several OUTPUTS may be defined at a time.

Synonyms:

CUT

Examples:

- OUTPUT OUT-432;
- OUTPUT PAYROLL-CHECK;
- OUT OUT-431;

ASSERT statement

OUTPUT section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value
[, name attribute-name attribute-value] ...;

Complementary Statements:
None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

OUTPUT section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [
{ integer } [ ,attr-name { attrv-name } ]
{ integer } [ { integer } ]
```

Complementary Statements:
none.

Usage Rules:

- It may be used in any section.
- A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CLASSIFICATION statement

OUTPUT section

Purpose:

To associate security CLASSIFICATION requirements with data in the target system.

Syntax:

```
CLASSIFICATION classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

CLS CLASSIFICATIONS

Examples:

- CLASSIFICATION IS PERSONNEL, SEC-LEVEL 3;
- CLS RING-LEVEL 2, UPDATE;

CONSISTS statement

OUTPUT section

Purpose:

To describe the combination of GROUPS, and/or ELEMENTS which make up this OUTPUT. This implies that each instance of the OUTPUT will contain values of the GROUP and ELEMENT names. A GROUP or ELEMENT may be repeated the number of times denoted by the SYSTEM-PARAMETER.

Syntax:

```

                                element-
CONSISTS OF [ system-parameter ] group-name
                                element-
                                [ , [ system-parameter ] group-name ] ... ;
```

Complementary Statements:

CONTAINED statement in a GROUP or ELEMENT section.

Usage Rules:

- The names, other than the system-parameters, must be GROUP or ELEMENT names.
- An OUTPUT may contain several GROUPS or ELEMENTS.

Synonyms:

CSTS

Examples:

- CONSISTS OF TWO DATA-GROUP-1;
- CONSISTS: DATA-GROUP-1, ELEMENT-A;
- CSTS OF SPAN-ELEMENT-A;
- CSTS: GROUP-NO-1, GROUP-NO-2;

CONTAINED statement

OUTPUT section

Purpose:

To give the SETS that contain this OUTPUT. An OUTPUT being contained in a SET means that the data values contained in the OUTPUT will be included in the logical SET.

Syntax:

CONTAINED IN set-name(s) ;

Complementary Statements:

CCNSISTS statement in SET section.

Usage Rules:

- The names must be SET names.
- Several SETS may contain a given OUTPUT.

Synonyms:

CNTD

Examples:

- CONTAINED IN MASTER-FILE;
- CNTD: HS-1,HS-2;
- CNTD FILE-1;

DERIVED statement

OUTPUT section

Purpose:

To give a PROCESS that DERIVES values for the OUTPUT and, optionally, the SETS, INPUTS, ENTITIES, GROUPS, and/or ELEMENTS used in the derivation.

Syntax:

```

DERIVED BY process-name(s) [
    [ group-                ]
    [ entity-               ]
    [ USING set-name(s)    ] ;
    [ input-                ]
    [ element-              ]

```

Complementary Statements:

DERIVES or USES statement in a PROCESS section and USED BY statement in a SET, INPUT, ENTITY, GROUP or ELEMENT section.

Usage Rules:

-Several PROCESSES may derive values for an OUTPUT.

Synonyms:

DFVD USG

Examples:

- DERIVED BY PROCESS-A USING INPUT-1;
- DERIVED BY PROCESS-1 USING ENTITY-A, ENTITY-B;
- DFVD PROCESS-Q USG INPUT-1;
- DFVD PROCESS-NAME USG ENTITY-A, GROUP-B;

DESCRIPTION statement

OUTPUT section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

GENERATED statement

OUTPUT section

Purpose:

To identify the PROCESS which is responsible for producing this OUTPUT.

Syntax:

GENERATED BY process-name(s) ;

Complementary Statements:

GENERATES statement in PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- An OUTPUT can be GENERATED by more than one PROCESS.

Synonyms:

GEND

Examples:

- GENERATED BY OUTPUT-PROCESS-1;
- GEND BY PROCESS-UPDATE;

HAPPENS statement

OUTPUT section

Purpose:

To give the volume for this OUTPUT. More than one instance of an OUTPUT may occur over some period of time. The number of instances of the OUTPUT which occur in a time INTERVAL is expressed with this statement.

Syntax:

HAPPENS system-parameter TIMES-PER interval-name ;

Complementary Statements:

None.

Usage Rules:

- The name must be an INTERVAL name.
- The statement may be given as many times as necessary for different INTERVALS.

Synonyms:

HAP TIMP

Examples:

- HAPPENS TWELVE TIMES-PER INT-A;
- HAP THREE TIMP INT-2;

KEYWORDS statement

OUTPUT section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIFS statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

PART statement

OUTPUT section

Purpose:

To show the structural relationship of this OUTPUT to a higher-level OUTPUT. This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

PART OF output-name ;

Complementary Statements:

SUBPARTS statement in an OUTPUT section.

Usage Rules:

-The name must be an OUTPUT name.

-Only one OUTPUT name can be given, hence, only a tree structure may be established.

Synonyms:

none.

Examples:

-PART OF OUTPUT-897;

RECEIVED statement

OUTPUT section

Purpose:

To show which INTERFACE uses or receives the OUTPUT.

Syntax:

RECEIVED BY interface-name(s) ;

Complementary Statements:

RECEIVES statement in INTERFACE section.

Usage Rules:

-The names must be INTERFACE names.

Synonyms:

RCVD

Examples:

- RECEIVED BY RWE-104;
- RCVD DEPT-89;

RESPONSIBLE-PROBLEM-DEFINER statement

OUTPUT section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

OUTPUT section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

OUTPUT section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

OUTPUT section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBPARTS statement

OUTPUT section

Purpose:

To show the structural relationship of this OUTPUT to lower-level OUTPUT(S). This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBPARTS ARE out put-name(s) ;

Complementary Statements:

PART statement in an OUTPUT section.

Usage Rules:

- The names must be OUTPUT names.
- An OUTPUT may be composed of several other OUTPUTS.

Synonyms:

SUBP

Examples:

- SUBPARTS ARE OUT-101, OUT-103;
- SUBP OUT-309, OUTPUT-897;

SYNONYMS statement

OUTPUT section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- The statement may be used in any section except a MEMO section, or a DEFINE section for a SYNONYM.
- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE O-11, OUTPUT-11;
- SYNONYM IS OUTPUT-11;
- SYN ALPHA;

TRACE-KEY statement

OUTPUT section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.13 PROBLEM-DEFINER section header statement

Purpose:

To define a PROBLEM-DEFINER or DEFINERS. The PROBLEM-DEFINER is the person responsible for one or more URL object definitions. This section identifies for which other sections within the documentation the PROBLEM-DEFINER has responsibility. This is useful in establishing good documentation controls for the system.

Syntax:

PROBLEM-DEFINER problem-definer-name(s) ;

Usage Rules:

- Must be the first statement in a PROBLEM DEFINER section.
- Several PROBLEM-DEFINERS may be defined at once.

Synonyms:

PD PROBLEM-DEFINERS

Examples:

- PROBLEM-DEFINER J-SURTES;
- PROBLEM-DEFINERS: P-REZK, J-SMITH;
- PD: E-WINTERS;

ASSERT statement

PROBLEM-DEFINER section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value
[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASPT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

PROBLEM-DEFINER section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [
{ integer } [ ,attr-name { attrv-name } ]
{ integer } ] .
```

Complementary Statements:
none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

DESCRIPTION statement

PROBLEM-DEFINER section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

```
DESCRIPTION ;  
    comment-entry ;
```

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

```
DESCRIPTION;  
    THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT  
    THIS SECTION TO DO;  
  
DESC;  
    ANY RELEVANT INFORMATION GOES HERE;
```

KEYWORDS statement

PROBLEM-DEFINER section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MAILBOX statement

PROBLEM-DEFINER section

Purpose:

To identify the location or address where this PROBLEM-DEFINER may be reached.

Syntax:

MAILBOX IS mailbox-name ;

Complementary Statements:

APPLIES statement in DEFINE section for a MAILBOX.

Usage Rules:

- The name must be a MAILBOX name.
- A PROBLEM-DEFINER may only have one MAILBOX.

Synonyms:

ECX MBX MAILBOXES

Examples:

- MAILBOX IS USERID-AA110;
- BOX IS FOUR-FORTY-FIVE-HAMILTON-AVE;
- MBX IS FIVE-WORLD-TRADE-CENTER;

RESPONSIBLE statement

PROBLEM-DEFINER section

Purpose:

To give the sections for which a PROBLEM-DEFINER is responsible.

Syntax:

RESPONSIBLE FOR name(s) ;

Complementary Statements:

RESPONSIBLE-PROBLEM-DEFINER statement.

Usage Rules:

-The names may be any type of name except a PROBLEM-DEFINER name or a MAILBOX name.

-Only one PROBLEM-DEFINER may be RESPONSIBLE for any section.

Synonyms:

RESP RES

Examples:

- RESPONSIBLE FOR P-101;
- RESP FOR P-10,P-11,P-12,P-13,P-14;

SECURITY statement

PROBLEM-DEFINER section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

PROBLEM-DEFINER section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

PROBLEM-DEFINER section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

PROBLEM-DEFINER section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- The statement may be used in any section except a MEMO section, or a DEFINE section for a SYNONYM.
- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE P-11, PROBLEM-DEFINER-11;
- SYNONYM IS PROBLEM-DEFINER-11;
- SYN ALPHA;

TRACE-KEY statement

PROBLEM-DEFINER section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.14 PROCESS section header statement

Purpose:

To allow a detailed description of a PROCESS or PROCESSES. This section is used to show how data is used within the target system. For instance, a PROCESS can validate INPUTS, produce OUTPUTS, store and manipulate data to meet the objectives of the system, and cause the initiation of additional PROCESS(ES). It is also used to show the structure of the system and its component subsystems.

Syntax:

PROCESS process-name(s) ;

Usage Rules:

- Must be the first statement in a PROCESS section.
- Several PROCESSES may be defined at once.

Synonyms:

PROC PEC

Examples:

- PROCESS P-101;
- PROC P-32, P-86;
- PROCESS P-789,P-539;

ASSERT statement

PROCESS section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

PROCESS section

Purpose:

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ]  
                        { integer } [ { integer } ]
```

Complementary Statements:
none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAP ZZZ9V9;

DERIVES statement

PROCESS section

Purpose:

To give the data which is DERIVED by this PROCESS , and, optionally, the data used to DERIVE it.

Syntax:

```

      set-      [      set-      ]
      output-   [      input-    ]
DERIVES element-name(s) [ USING element-name(s) ] ;
      entity-   [      entity-   ]
      group-    [      group-    ]

```

Complementary Statements:

DERIVED or USED BY statements in SET, ELEMENT, ENTITY, GROUP, or OUTPUT sections and USES statement in PROCESS section.

Usage Rules:

-A single PROCESS may DERIVE several different SETS, OUTPUTS, ELEMENTS, ENTITIES, or GROUPS.

Synonyms:

DEVS USG

Examples:

- DERIVES ELEMENT-407-X USING ELEMENT-407-Y;
- DERIVES ELEMENT-147 USING ELEMENT-48, ELEMENT-49, ELEMENT-50;
- DEVS ELE-22 USG ELE-221;
- DEVS ELE-186 USG ELE-1, ELE-17, ELE-23;

DESCRIPTION statement

PROCESS section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

GENERATES statement

PROCESS section

Purpose:

To give those OUTPUTS which are GENERATED by this PROCESS.

Syntax:

GENERATES output-name(s) ;

Complementary Statements:

GENERATED statement in OUTPUT section.

Usage Rules:

-The names must be OUTPUT names.

Synonyms:

GENS

Examples:

- GENERATES FIRST-OUTPUT;
- GENERATES OUTPUT-1, OUTPUT-2;
- GENS OUT-A;
- GENS OUT-A,OUT-B;

HAPPENS statement

PROCESS section

Purpose:

To give the number of times the PROCESS is used per INTERVAL. More than one instance of a PROCESS may occur over some period of time. The number of instances of the PROCESS which occur in a time INTERVAL is expressed with this statement.

Syntax:

HAPPENS system-parameter TIMES-PER interval-name ;

Complementary Statements:

None.

Usage Rules:

- The name must be an INTERVAL name.
- The statement may be given as many times as necessary for different INTERVALS.

Synonyms:

HAP TIMP

Examples:

- HAPPENS SIX TIMES-PER NEW-INTERVAL;
- HAP ONE TIMP OLD-DATE-INT;

INCEPTION-CAUSES statement

PROCESS section

Purpose:

To link an EVENT or EVENTS to the inception of the PROCESS .

Syntax:

INCEPTION-CAUSES event-name(s) ;

Complementary Statements:

INCEPTION statement in an EVENT section.

Usage Rules:

- The names must be EVENT names.
- A PROCESS may initiate several EVENTS.

Synonyms:

INCC

Examples:

- INCEPTION-CAUSES UPDATE-EVT;
- INCC EVENT-1,EVENT-2;

INTERRUPTED statement

PROCESS section

Purpose:

To specify an EVENT/EVENTS, INPUT/INPUTS, or PROCESS/PROCESSES which interrupt this PROCESS. Also, to specify CONDITIONS for which changes of state will cause interruption of this PROCESS.

Syntax:

```
event-  
INTERRUPTED BY input-name(s) ;  
process-
```

```
INTERRUPTED WHEN condition-name BECOMES { TRUE }  
                                           { FALSE }
```

Complementary Statements:

INTERRUPTS statement in the EVENT, INPUT, and PROCESS sections,
and BECOMING INTERRUPTS statement in the CONDITION section.

Usage Rules:

- A PROCESS may be INTERRUPTED by several EVENTS, INPUTS, or PROCESSES.
- Only one CONDITION may be specified in a single statement. Separate statements are required for each CONDITION.

Synonyms:

INTD

Examples:

- INTERRUPTED BY PURCHASE-ORDER-DELAY;
- INTD HIGH-PRIO-INPUT, NEW-TASK-INPUT;
- INTERRUPTED WHEN END-OF-FILE BECOMES FALSE ;
- INTD WHEN MACHINE-BREAKDOWN T;

INTERRUPTS statement

PROCESS section

Purpose:

To specify PROCESS(ES) which are interrupted by this PROCESS.

Syntax:

INTERRUPTS process-name(s);

Complementary Statements:

INTERRUPTED statement in the PROCESS section.

Usage Rules:

- A PROCESS may INTERRUPT several other PROCESSES.

Synonyms:

INTS

Examples:

- INTERRUPTS SUBPROCESS-A, SUBPROCESS-B;
- INTS SWITCHING-OPERATION;

KEYWORDS statement

PROCESS section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

-KEY ON-LINE PROCESS;

-KEYWORD TERMINAL;

MAINTAINS statement

PROCESS section

Purpose:

To give the RELATIONS and SUBSETTING-CRITERIA which are MAINTAINED by this PROCESS.

Syntax:

relation-
MAINTAINS subsetting-criteria-name(s) ;

Complementary Statements:

MAINTAINED statement in DEFINE section for SUBSETTING-CRITERION,
and MAINTAINED statement in RELATION section.

Usage Rules:

- The names must be either RELATION or SUBSETTING-CRITERIA names.
- A PROCESS may MAINTAIN several RELATIONS and SUBSETTING-CRITERIA.

Synonyms:

MTNS

Examples:

- MAINTAINS RELATION-SET;
- MTNS FIRST-RELATION, FIFTY-FIRST-SET;

MAKES statement

PROCESS section

Purpose:

To give CONDITION(S) whose states are set by this PROCESS.

Syntax:

```
MAKES condition-name(s) { TRUE }  
                        { FALSE } ;
```

Complementary Statements:

MADE statement in the CONDITION section.

Usage Rules:

- A PROCESS may MAKE several CCNDITIONS become either TRUE or FALSE.
- A PROCESS cannot MAKE some CCNDITIONS TRUE and some CONDITIONS FALSE in a single statement. Separate statements are required.

Synonyms:

MAK

Examples:

- MAKES PROCESS-COMPLETION TRUE;
- MAK INPUT-READ, PRODUCTION-BEGAN P ;

PART statement

PROCESS section

Purpose:

To show the structural relationship of this PROCESS to a higher-level PROCESS. This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

PART OF process-name ;

Complementary Statements:

SUBPARTS statement in a PROCESS section.

Usage Rules:

- The name must be a PROCESS name.
- Only one PROCESS name may be given, hence, only a tree structure can be established.

Synonyms:

none.

Examples:

-PART OF PAYROLL-SYSTEM;

PERFORMED statement

PROCESS section

Purpose:

To give the PROCESSOR that performs the PROCESS.

Syntax:

PERFORMED BY processor-name ;

Complementary Statements:

PERFORMS statement in PROCESSOR section.

Usage Rules:

- Only one PROCESSOR name may be given.

Synonyms:

PFMD

Examples:

- PERFORMED BY CPU-1;
- PFMD PROCESSOR-NO-1;

PROCEDURE statement

PROCESS section

Purpose:

To describe the sequence of operations needed to implement this PROCESS.

Syntax:

```
PROCEDURE ;  
    comment-entry ;
```

Complementary Statements:

None.

Usage Rules:

-Only one PROCEDURE statement may be given for any PROCESS.

Synonyms:

PRCD PRD

Examples:

- PROCEDURE;
 - 1. READ THE DATA FROM THE FILE
 - 2. CHECK TRANSACTION CODE
 - 3. CALL APPROPRIATE TRANSACTION PROCESS;
- PRCD;
 - ANY RELEVANT COMMENTS TO AID THE PROGRAM DESIGNER;

RECEIVES statement

PROCESS section

Purpose:

To give the INPUTS RECEIVED by this PROCESS.

Syntax:

RECEIVES input-name(s) ;

Complementary Statements:

RECEIVED statement in INPUT section.

Usage Rules:

- The names must be INPUT names.
- A PROCESS may RECEIVE more than one INPUT.

Synonyms:

RCVS

Examples:

- RECEIVES INPUT-100;
- RECEIVES INPUT-4A, INPUT-4B;
- RCVS INPUT-A100;

RESPONSIBLE-PROBLEM-DEFINER statement

PROCESS section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

RESOURCE-USAGE statement

PROCESS section

Purpose:

To give a pair of resource-usage parameter and resource usage parameter value for the PROCESS.

Syntax:

RESOURCE-USAGE :

system-parameter FOR resource-usage-parameter-name;

Complementary Statements:

RESOURCE-USAGE-PARAMETER-VALUE statement in RESOURCE-USAGE-PARAMETER section.

Usage Rules:

- The second term (system-parameter or number) is called the "resource-usage-parameter-value" (rup-value) for the resource-usage-parameter. A PROCESS may have several pairs of resource-usage-parameter-values as long as the resource usage parameters are not the same.

Synonyms:

RU

Examples:

- RESOURCE-USAGE: 10 FOR COMPLEXITY-RATING;
- RU 2000 FOR STATEMENTS-IN-PL;
- RU MAXIMUM-RATING RATING;

SECURITY statement

PROCESS section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

AD-A060 684 MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 F19628-76-C-0197

MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 F19628-76-C-0197

ESD-TR-78-129-VOL-2 NL

NL

AD
A060684

1000

SECURITY-ACCESS-RIGHT statement

PROCESS section

Purpose:

To give the type and level of security associated with a PROCESS during operation of the target system.

Syntax:

SECURITY-ACCESS-RIGHT classification-name [integer]
[, classification-name [integer]]... ;

Complementary Statements:

Ncne.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

SAR SECURITY-ACCESS-RIGHTS

Examples:

- SECURITY-ACCESS-RIGHTS ARE PERSONNEL, SEC-LEVEL 3;
- SAR RING-LEVEL 2, UPDATE;

SEE-MEMO statement

PROCESS section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGP-109;
- SM EPR-37, EPR-38;

SOURCE statement

PROCESS section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBPARTS statement

PROCESS section

Purpose:

To show the structural relationship of this PROCESS to lower-level PROCESS(ES). This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBPARTS ARE process-name(s) ;

Complementary Statements:

PAPT statement in a PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- A PROCESS may be composed of several other PROCESSES.

Synonyms:

SUBP

Examples:

- SUBPARTS ARE P-101, P-103;
- SUBP P-309, INPUT-EDIT-PROCESS;

SYNONYMS statement

PROCESS section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE P-11, PROCESS-11;
- SYNONYM IS PROCESS-11;
- SYN ALPHA;

TERMINATED statement

PROCESS section

Purpose:

To specify EVENT(S), INPUT(S), and/or PROCESS(ES) which terminate this PROCESS. Also, to specify CONDITIONS for which changes of state will terminate this PROCESS.

Syntax:

event-
TERMINATED BY input-name(s) ;
process-

TERMINATED WHEN condition-name BECOMES { TRUE }
{ FALSE } ;

Complementary Statements:

TERMINATES statement in the EVENT, INPUT, and PROCESS sections,
and BECOMING TERMINATES statement in the CONDITION section.

Usage Rules:

- A PROCESS may be TERMINATED by several EVENTS, INPUTS, or PROCESSES.
- Only one CONDITION may be specified in a single statement. Separate statements are required for each CONDITION.

Synonyms:

TRMD

Examples:

- TERMINATED BY END-OF-INPUT;
- TRMD BY LAST-INPUT, NEW-ORDER-INPUT;
- TRMD ERROR-PROC, SEARCH-PROC;
- TRMD WHEN FATAL-ERROR BECOMES FALSE;

TERMINATES statement

PROCESS section

Purpose:

To specify a PROCESS/PROCESSES that are terminated by this PROCESS.

Syntax:

TERMINATES process-name(s) ;

Complementary Statements:

TERMINATED statement in PROCESS section.

Usage Rules:

- A PROCESS may TERMINATE several other PROCESSES.

Synonyms:

TFMS

Examples:

- TERMINATES OUTPUT-PRODUCTION;
- TFMS SET-UP-PROC, ERROR-CHECKING;

TERMINATION-CAUSES statement

PROCESS section

Purpose:

To indicate which EVENT or EVENTS occur when this PROCESS finishes.

Syntax:

TERMINATION-CAUSES event-name(s) ;

Complementary Statements:

TERMINATION statement in an EVEN. n.

Usage Rules:

- The names must be EVENT names.
- A PROCESS may terminate several different EVENTS.

Synonyms:

TERC

Examples:

- TERMINATION-CAUSES UPDATE-EVENT;
- TERC ISSUE-CHECK-EVENT;

TRACE-KEY statement

PROCESS section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

TRIGGERED statement

PROCESS section

Purpose:

To give the EVENT/EVENTS, INPUT/INPUTS, and PROCESS/PROCESSES which can TRIGGER this PROCESS. Also, to specify a CONDITION which may trigger this PROCESS.

Syntax:

```
event-  
TRIGGERED BY input-name(s) ;  
process-
```

```
TRIGGERED WHEN condition-name BECOMES { TRUE }  
                                           { FALSE } ;
```

Complementary Statements:

TRIGGERS statement in EVENT, INPUT, and PROCESS sections, and BECOMING TRIGGERS statement in the CONDITION section.

Usage Rules:

- Only one CONDITION may be specified in a single statement. A separate statement is necessary for each CONDITION specified.
- Several triggering EVENTS, INPUTS, or PROCESSES may be given.

Synonyms:

TFGD

Examples:

- TRIGGERED BY UPDATE-EVENT;
- TFGD ORDER-PROC, ERROR-CHECKING, INFO-RETRIEVAL-PROC;
- TRIGGERED WHEN DATA-FOUND BECOMES TRUE;

TRIGGERS statement

PROCESS section

Purpose:

To specify a PROCESS/PROCESSES which are triggered by this PROCESS.

Syntax:

TRIGGERS process-name(s) ;

Complementary Statements:

TRIGGERED statement in the PROCESS section.

Usage Rules:

- A PROCESS may TRIGGER several other PROCESSES.

Synonyms:

TRGS

Examples:

- TRIGGERS MAIN-PROCESSING;
- TRGS INPUT-CHECKING, MAIN-PROCESSING;

UPDATES statement

PROCESS section

Purpose:

To give the ENTITIES, GROUPS, ELEMENTS and/or SETS which are updated by this PROCESS.

Syntax:

```

      group-      [      group-      ]
      entity-     [      entity-     ]
UPDATES element-name(s) [ USING element- name(s) ] :
      set-        [      set-        ]
                  [      input-       ]
```

Complementary Statements:

UPDATED or USED BY statements in ENTITY, GROUP, ELEMENT and SET sections and USES statement in PROCESS section.

Usage Rules:

none.

Synonyms:

UPDS USG

Examples:

- UPDATES HS-SEGMENT, HT-SEGMENT;
- UPDS AQ-SEGMENT USING E-2, E-5;

USES statement

PROCESS section

Purpose:

To give those SETS, GROUPS, ELEMENTS, INPUTS and ENTITIES used by the PROCESS.

Syntax:

```

                set-           [           set-           ]
                input-         [           *output-        ]
USES element-name(s) [ TO { DERIVE } element- name(s) ] ;
                group-         [           { UPDATE } group-   ]
                entity-        [           entity-          ]

```

* Output-name(s) may only be used with the DERIVE clause.

Complementary Statements:

USED, UPDATED or DERIVED statement in a SET, GROUP, ELEMENT or ENTITY section and DERIVES or UPDATES statement in PROCESS section.

Usage Rules:

- A PROCESS may use several different SETS, GROUPS, ELEMENTS, INPUTS or ENTITIES.

Synonyms:

DFV UPD

Examples:

- USES TASK-FILE;
- USES PERSONNEL-FILE, PAYROLL-FILE;

UTILIZED statement

PROCESS section

Purpose:

To show the structural relationship of this PROCESS to higher-level PROCESSES. This statement allows PROCESSES to be used by more than one higher-level PROCESS.

Syntax:

UTILIZED BY process-name(s) ;

Complementary Statements:

UTILIZES statement in the PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- A PROCESS may be UTILIZED by several PROCESSES

Synonyms:

UTLD

examples:

- UTILIZED LP-ALGORITHM;
- UTILIZED COMMON-INPUT-PROCESS, COMMON-OUTPUT-PROCESS;
- UTLD: TAPE-READ-PROCESS;
- UTLD: UPDATE-BILL-PROC-1, UPDATE-BILL-PROC-2;

UTILIZES statement

PROCESS section

Purpose:

To show the structural relationship of this PROCESS to lower-level PROCESSES. This statement allows several higher-level PROCESSES to share the use of the same lower-level PROCESS.

Syntax:

UTILIZES process-name(s) ;

Complementary Statements:

UTILIZED statement in the PROCESS section.

Usage Rules:

- The names must be PROCESS names.
- A PROCESS may UTILIZE several PROCESSES

Synonyms:

UTLS

Examples:

- UTILIZES LP-ALGORITHM;
- UTILIZES COMMON-INPUT-PROCESS, COMMON-OUTPUT-PROCESS;
- UTLS: TAPE-READ-PROCESS;
- UTLS: UPDATE-BILL-PROC-1, UPDATE-BILL-PROC-2;

4.15 PROCESSOR section header statement

Purpose:

To allow a detailed description of a PROCESSOR.

Syntax:

PROCESSOR processor-name(s);

Usage Rules:

- Must be the first statement in a PROCESSOR section.
- More than one PROCESSOR may be defined at once.

Synonyms:

PROCR PRCR PROCESSORS

Examples:

- PROCESSOR PR-1;
- PRCR CPU, DISK-MEMORY;

ASSERT statement

PROCESSOR section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

Ncne.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

PROCESSOR section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ] ..  
                        { integer } [ { integer } ]
```

Complementary Statements:
none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CONSUMES statement

PROCESSOR section

Purpose:

To give the resource consumption value for the PROCESSOR.

Syntax:

CONSUMES resource-name AT RATE OF
system-parameter PER resource-usage-parameter-name;

Complementary Statements:

CONSUMED statement in RESOURCE section.

Usage Rules:

- A name may have several CONSUMES statements as long as they are not contradictory, i.e. , at most one CONSUMED statement is allowed for a unique pair of resource-name and resource-usage-parameter-name.

Synonyms:

CNSS

Examples:

- CONSUMES REAL TIME AT A RATE OF 10 PER NUMBER-OF-CHARACTERS;
- CNSS DOLLARS RATE X PER DIFFICULTY-GRADING;

DESCRIPTION statement

PROCESSOR section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

PROCESSOR section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

PART statement

PROCESSOR section

Purpose:

To show the structural relationship of this PROCESSOR to a higher level PROCESSOR. This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

PART OF processor-name;

Complementary Statements:

SUBPARTS statement in PROCESSOR section.

Usage Rules:

- Only one PROCESSOR name may be given, hence only a tree structure can be established.

Synonyms:

NONE.

Examples:

- PART OF MACHINES;

PERFORMS statement

PROCESSOR section

Purpose:

To give the PROCESSES that the PROCESSOR performs.

Syntax:

PERFORMS process-name(s) ;

Complementary Statements:

PERFORMED statement in PROCESS section.

Usage Rules:

- More than one PROCESS may be performed by a PROCESSOR, but a PROCESS may be performed by one PROCESSOR only.

Synonyms:

PFMS

Examples:

- PERFORMS PAYFOII-PROCESSING;
- PFMS PROCESS-A, PROCESS-B;

RESPONSIBLE-PROBLEM-DEFINER statement

PROCESSOR section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules: •

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

PROCESSOR section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-OPMISTON, S-MENNEL;
- SEC L-HANNON;

SECURITY-ACCESS-RIGHT statement

PROCESSOR section

Purpose:

To give the type and level of security associated with a PROCESSOR during operation of the target system.

Syntax:

SECURITY-ACCESS-RIGHT classification-name [integer]
[, classification-name [integer]]... ;

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

SAR SECURITY-ACCESS-RIGHTS

Examples:

- SECURITY-ACCESS-RIGHTS ARE PERSONNEL, SEC-LEVEL 3;
- SAR RING-LEVEL 2, UPDATE;

SEE-MEMO statement

PROCESSOR section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

PROCESSOR section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SUBPARTS statement

PROCESSOR section

Purpose:

To show the structural relationship of this PROCESSOR to lower-level PROCESSORS. This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBPARTS ARE processor-name(s) ;

Complementary Statements:

PART statement in PROCESSOR section.

Usage Rules:

- A PROCESSOR may be composed of several other PROCESSORS.

Synonyms:

SUBP

Examples:

- SUBPARTS ARE HUMAN, MACHINES;
- SUBP PR-1, PR-2, PR-3;

SYNONYMS statement

PROCESSOR section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE P-11, PROCESSOR-11;
- SYNONYM IS PROCESSOR-11;
- SYN ALPHA;

TRACE-KEY statement

PROCESSOR section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.16 RELATION section header statement

Purpose:

To define a RELATION or RELATIONS. This section shows how two ENTITIES are logically connected. Examples of relations are husband-to-wife or employee-to-company.

Syntax:

RELATION relation-name(s) ;

Usage Rules:

- Must be the first statement of every RELATION section.
- Several RELATIONS may be defined at once.

Synonyms:

RLN RELATIONS

Examples

- RELATION NH-RELATION;
- RLN NI-RELATION, NS-RELATION;
- RELATIONS REL-1, REL-2, REL-3;

ASSERT statement

RELATION section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

Ncne.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ASSOCIATED-DATA statement

RELATION section

Purpose:

To give those GROUPS and/or ELEMENTS which are the result of the RELATION being described or which describe the RELATION. Although the data may be contained in either or both ENTITIES. ASSOCIATED-DATA does not belong to either ENTITY RELATION being described. ASSOCIATED DATA does not belong to either ENTITY exclusively, but to both jointly.

Syntax:

group-
ASSOCIATED-DATA IS element-name(s) ;

Complementary Statements:

ASSOCIATED statement in ELEMENT and GROUP section.

Usage Rules:

- The names must be either ELEMENT or GROUP names.
- The ELEMENTS associated with a RELATION may not be part of an ENTITY.

Synonyms:

ASCD

Examples:

- ASSOCIATED-DATA IS SPAN-SEGMENT;
- ASSOCIATED-DATA IS ELE-1,ELE-2,GROUP-9;
- ASCD LINK-SEGMENT;
- ASCD ELEMENT-A, GROUP-9;

ATTRIBUTES statement

RELATION section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ { attrv-name } ]  
                                { integer } [ ,attr-name { integer } ]
```

Complementary Statements:

none.

Usage Rules:

- It may be used in any section.
- A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

BETWEEN statement

RELATION section

Purpose:

To give the ENTITIES which are related, e.g. Logically connected, via a particular RELATION.

Syntax:

BETWEEN entity-name AND entity-name ;

Complementary Statements:

RELATED statement in ENTITY section.

Usage Rules:

- Both names must be ENTITY names, they may, however, be the same ENTITY name.
- All RELATIONS are binary.
- All RELATIONS must have exactly one BETWEEN statement which gives the ENTITIES involved in the RELATION.

Synonyms:

BTWN

Examples:

- BETWEEN WOMAN AND MAN;
- BETWEEN ENTITY-1 AND ENTITY-2 ;
- BETWEEN RECORD-1 AND RECORD-2;
- BTWN EMP-INFO JCB-INFO ;

CARDINALITY statement

RELATION section

Purpose:

To define the number of times this RELATION applies in the system.

Syntax:

CARDINALITY IS system-parameter ;

Complementary Statements:

None.

Usage Rules:

-A RELATION may have only one CARDINALITY.

Synonyms:

CARD OCCS OCCURRENCES

Examples:

- CARDINALITY IS TWENTY;
- CARD FORTY-SEVEN;

CONNECTIVITY statement

RELATION section

Purpose:

To define the number of occurrences in the RELATION of one ENTITY with respect to the other. For example, one could specify that there is one company-entity related to many employee-entities.

Syntax:

CONNECTIVITY IS system-parameter TO system-parameter ;

Complementary Statements:

None.

Usage Rules:

- Any RELATION may have only one CONNECTIVITY given.

Synonyms:

CCNN

Examples:

- CONNECTIVITY IS ONE TO ONE;
- CONN MANY TO TWO;

DERIVATION statement

RELATION section

Purpose:

To give the DERIVATION rules for those RELATIONS which are derivable for the data. This implies that the RELATION being described is a DERIVED RELATION, not a direct RELATION.

Syntax:

DERIVATION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DEVN

Examples:

DERIVATION;

THIS RELATIONSHIP EXISTS TO SHOW HOW UPON ENTRY OF THE TIME CARD AN UPDATE OCCURS;

DEVN;

ANY RELEVANT COMMENTS MAY BE ENTERED;

DESCRIPTION statement

RELATION section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

Ncne.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

RELATION section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MAINTAINED statement

RELATION section

Purpose:

To designate those PROCESSES which change the instances of the ENTITIES

Syntax:

MAINTAINED BY process-name(s) ;

Complementary Statements:

MAINTAINS statement in PROCESS section.

Usage Rules:

- The names must be process- names.
- A RELATION may be MAINTAINED BY more than one PROCESS.

Synonyms:

MTND

Examples:

- MAINTAINED BY process-6543;
- MTND p-18, p-190;

RESPONSIBLE-PROBLEM-DEFINER statement

RELATION section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

RELATION section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

RELATION section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

RELATION section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- It may be used in any section except a DEFINE section for a SOURCE.
- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

RELATION section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:

DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE R-11, RELATION-11;
- SYNONYM IS RELATION-11;
- SYN ALPHA;

TRACE-KEY statement

RELATION section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.17 RESOURCE section header statement

Purpose:

To allow a detailed description of the contents of a RESOURCE.
A RESOURCE is something that is consumed by the target system.
It is used in the target system to model system performance.

Syntax:

RESOURCE resource-name(s);

Usage Rules:

- It must be the first statement in a RESOURCE section.
- Several RESOURCES may be defined at once.

Synonyms:

RSC

Examples:

- RESOURCE CPU-TIME, MAN-POWER;
- RSC MONEY;

ASSERT statement

RESOURCE section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

RESOURCE section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ] [ integer ] [ integer ] .
```

Complementary Statements:

none.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CONSUMED statement

RESOURCE section

Purpose:

To give the names of PROCESSORS that consume the RESOURCE.

Syntax:

CONSUMED BY processor-name(s) AT RATE OF
system-parameter PER resource-usage-parameter-name;

Complementary Statements:

CONSUMES statement in PROCESSOR section.

Usage Rules:

- More than one processor-name may be specified.

Synonyms:

CNSD

Examples:

- CONSUMED BY CPU AT A RATE OF 100,000 PER MINUTE;
- CNSD PROCESSOR-A, PROCESSOR-B RATE 9000 PER JOB;

DESCRIPTION statement

RESOURCE section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

RESOURCE section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MEASURED statement

RESOURCE section

Purpose:

To give the UNIT name that the RESOURCE is measured in.

Syntax:

MEASURED IN unit-name;

Complementary Statements:

MEASURES statement in UNIT section.

Usage Rules:

- A RESOURCE may be measured in only one UNIT.

Synonyms:

MSRD

Examples:

- MEASURED IN DOLLARS;
- MSRD MILLI-SECONDS;

RESPONSIBLE-PROBLEM-DEFINER statement

RESOURCE section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RFD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

RESOURCE section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SFC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

RESOURCE section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

RESOURCE section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

RESOURCE section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE F-11, RESOURCE-11;
- SYNONYM IS RESOURCE-11;
- SYN ALPHA;

TRACE-KEY statement

RESOURCE section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.18 RESOURCE-USAGE-PARAMETER section header statement

Purpose:

To allow a detailed description of RESOURCE-USAGE-PARAMETER(S).

Syntax:

RESOURCE-USAGE-PARAMETER resource-usage-parameter-name(s) ;

Usage Rules:

- Must be the first statement in a RESOURCE-USAGE-PARAMETER section.
- More than one RESOURCE-USAGE-PARAMETER may be defined at once.

Synonyms:

RUP

Examples:

- RESOURCE-USAGE-PARAMETER RUP-1;
- RUP DIFFICULTY-GRADING;

ASSERT statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

```

ASSERT name attribute-name attribute-value
        [, name attribute-name attribute-value] ...;
    
```

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSEPT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ] ..  
                        { integer } [ { integer } ]
```

Complementary Statements:
ncne.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

DESCRIPTION statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT
THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RESOURCE-USAGE-PARAMETER-VALUE statement RESOURCE-USAGE-PARAMETER
section

Purpose:

To give the resource-usage-parameter-value (rup value) for the pair of RESOURCE-USAGE-PARAMETER and process.

Syntax:

RESOURCE-USAGE-PARAMETER-VALUE :

system-parameter FOR process-name;

Complementary Statements:

RESOURCE-USAGE statement in PROCESS section.

Usage Rules:

- There may be at most one RESOURCE-USAGE-PARAMETER-VALUE for each unique pair of RESOURCE-USAGE-PARAMETER and PROCESS.

Synonyms:

RUP-VALUE RUPV

Examples:

- RESOURCE-USAGE-PARAMETER-VALUE:
10 FOR PROCESS-1;
- RUPV MAX-RATING PAYROLL-PROCESSING;

RESPONSIBLE-PROBLEM-DEFINER statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which
he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section,
hence, this statement may only be used once per section.

Synonyms:

RPD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-ORMISTON, S-MENNEL;
- SEC L-HANNON;

SEE-MEMO statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name (s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE P-11, RESOURCE-USAGE-PARAMETER-11;
- SYNONYM IS RESOURCE-USAGE-PARAMETER-11;
- SYN ALPHA;

TRACE-KEY statement

RESOURCE-USAGE-PARAMETER
section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

4.19 SET section header statement

Purpose:

To allow a detailed description of a SET. For example, this section allows the PROBLEM-DEFINER to show how ENTITIES defined within the system are collected together for information processing purposes. SETS can be defined as physical or logical views of the data as seen by the user, designer, and/or programmer.

Syntax:

SET set-name(s) ;

Usage Rules:

- It must be the first statement in the SET section.
- Several SETS may be defined at a time.

Synonyms:

none.

Examples:

- SET FORECAST-INFO;
- SET TRANSACTION-INFO ;

ASSERT statement

SET section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

None.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASRT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

SET section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ] ...  
                        { integer } [ { integer } ]
```

Complementary Statements:

none.

Usage Rules:

- It may be used in any section.
- A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

CARDINALITY statement

SET section

Purpose:

To define the number of times this SET appears in the system.

Syntax:

CARDINALITY IS system-parameter ;

Complementary Statements:

None.

Usage Rules:

-A SET may have only one CARDINALITY.

Synonyms:

CARD OCCS OCCURRENCES

Examples:

- CARDINALITY IS TEN;
- CARD FORTY-SEVEN;

CLASSIFICATION statement

SET section

Purpose:

To associate security CLASSIFICATION requirements with data in the target system.

Syntax:

```
CLASSIFICATION classification-name [ integer ]  
[ , classification-name [ integer ] ]... ;
```

Complementary Statements:

None.

Usage Rules:

- The name must be a CLASSIFICATION name.

Synonyms:

CLS CLASSIFICATIONS

Examples:

- CLASSIFICATION IS PERSONNEL, SEC-LEVEL 3;
- CLS RING-LEVEL 2, UPDATE;

CONSISTS statement

SET section

Purpose:

To describe the combination of INPUTS, OUTPUTS, and ENTITIES which make up this SET. This implies that each instance of the SET will contain values of the INPUT, OUTPUT and ENTITY names. An INPUT, OUTPUT or ENTITY may be repeated the number of times denoted by the SYSTEM-PARAMETER.

Syntax:

```
CONSISTS OF [ system-parameter ] input-  
output-name  
entity-  
  
[ , [ system-parameter ] input-  
output-name ] ... ;  
entity-
```

Complementary Statements:

CONTAINED statement in an ENTITY, INPUT or OUTPUT section.

Usage Rules:

- The names must be ENTITY, INPUT or OUTPUT names.
- A SET may contain several INPUTS, OUTPUTS, and ENTITIES.

Synonyms:

CSTS

Examples:

- CONSISTS OF DATA-ENTITY-1;
- CONSISTS OF: DATA-ENTITY-1, DATA-ENTITY-2;
- CSTS: ABSTRACT-1, ABSTRACT-2;

DERIVATION statement

SET section

Purpose:

To express the specific system actions necessary to obtain the correct SET. This statement contains rules for DERIVATION which can be the DERIVED BY USING clause in the SET section.

Syntax:

DERIVATION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DEVN

Examples:

- DERIVATION;
THIS SET OF INFORMATION WAS DERIVED FROM THE PAYROLL FILES TO
THE OLD PAYSYSYEM;

DERIVATION;

RULES FOR ADDITION:

ITEM MASTER-A ADDED WITH A TRANSACTION-CODE-74;

DERIVED statement

SET section

Purpose:

To give a PROCESS that DERIVES values for the SET and the SETS, INPUTS, ENTITIES, GROUPS, and/or ELEMENTS used in the DERIVATION.

Syntax:

```

DERIVED BY process-name(s) [ USING
                             [ group-      ]
                             [ entity-     ]
                             [ set-name(s) ] ;
                             [ input-      ]
                             [ element-    ]

```

Complementary Statements:

DERIVES or USES statement in a PROCESS section and USED BY statement in a SET, INPUT, ENTITY, GROUP or ELEMENT section.

Usage Rules:

-Several PROCESSES may DERIVE values for a SET.

Synonyms:

DEVD USG

Examples:

- DERIVED BY PROCESS-A USING INPUT-1;
- DERIVED BY PROCESS-1 USING ENTITY-A, ENTITY-B;
- DEVD PROCESS-2 USG INPUT-1;
- DEVD PROCESS-NAME USG ENTITY-A, GROUP-B;

DESCRIPTION statement

SET section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:
None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT
THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

SET section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

RESPONSIBLE-PROBLEM-DEFINER statement

SET section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- It may be used in any section except the PROBLEM-DEFINER section.
- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RFD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

RESPONSIBLE-INTERFACE statement

SET section

Purpose:

To give the INTERFACE which is responsible for this SET.

Syntax:

RESPONSIBLE-INTERFACE IS interface-name(s) ;

Complementary Statements:

RESPONSIBLE FOR in the INTERFACE section.

Usage Rules:

-The names must be INTERFACE names.

Synonyms:

RINT

Examples:

- RESPONSIBLE-INTERFACE IS PAYROLL-SYSTEM;
- RINT: ENGINEERING-DEPT;

SECURITY statement

SET section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SFC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-OFMISTON, S-MENNEL;
- SFC L-HANNON;

SEE-MEMO statement

SET section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

SET section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SFC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;
- SOURCES ARE SDP-3-1,SDP-3-2,MEMO-23-MAY-1974;

SUBSET statement

SET section

Purpose:

To show the structural relationship of this SET to higher-level SET(S). This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBSET OF set-name(s) ;

Complementary Statements:

SUBSETS statement in SET section.

Usage Rules:

- The names in name(s) must be SET names.
- A SET may be a SUBSET of several other SETS.

Synonyms:

SST

Examples:

- SUBSET OF SET-GROUP-BANKS, SET-GROUP-CKTS;
- SST: STUDENT-INFO, COURSE-INFO;

SUBSETS statement

SET section

Purpose:

To show the structural relationship of this SET to lower-level SET(S). This statement can be used to express a top-down or bottom-up view of the system.

Syntax:

SUBSETS ARE set-name(s) ;

Complementary Statements:

SUBSET statement in a SET section.

Usage Rules:

- The names must be SET names.
- Many SETS may be SUBSETS to one SET.

Synonyms:

SSTS

Examples:

- SUBSETS ARE SET-GROUP-BANKS, SET-GROUP-CKTS;
- SSTS: STUDENT-INFO, COURSE-INFO;

SUBSETTING-CRITERIA statement

SET section

Purpose:

To indicate what data and/or rules are to be used to extract a portion of the data from the SET.

Syntax:

group-
SUBSETTING-CRITERIA ARE element-name(s) ;
subsetting-criterion-

Complementary Statements:

APPLIES statement in DEFINE section for SUBSETTING-CRITERION,
and SUBSETTING-CRITERION statement in ELEMENT and GROUP
sections.

Usage Rules:

- The names must be either ELEMENT or GROUP names.
- If the SUBSETTING-CRITERIA is an ELEMENT or a GROUP then it must be part of the ENTITY which is a legal member of this SET.
- A SET may have more than one SUBSETTING-CRITERIA.
- If a GROUP is given for the SUBSETTING-CRITERIA then the ELEMENTS which make up the GROUP taken together form the SUBSETTING-CRITERIA.

Synonyms:

SSCA

Examples:

- SUBSETTING-CRITERIA ARE GROUP-BANKS, GROUP-CKTS;
- SSCA: GROUP-107, GROUP-108;

SYNONYMS statement

SET section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- The statement may be used in any section except a MEMO section, or a DEFINE section for a SYNONYM.
- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE S-11, SET-11;
- SYNONYM IS SET-11;
- SYN ALPHA;

TRACE-KEY statement

SET section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

UPDATED statment

SET section

Purpose:

To indicate those PROCESSES which UPDATE this SET, and optionally, to specify the data used to do the UPDATING.

Syntax:

```

        [      group-      ]
        [      entity-     ]
    UPDATED BY process-name(s) [ USING element- name(s) ] ;
        [      input-      ]
        [      set-        ]

```

Complementary Statements:

UPDATES or USES statement in PROCESS section and USED BY statement in INPUT, SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-A SET may be UPDATED by several different PROCESSES.

Synonyms:

UPDD USG

Examples:

- UPDATED BY INPUT-PROCESS;
- UPED PROC-1, PROC-2, PROC-789;

USED statement

SET section

Purpose:

To indicate the PROCESS(ES) that USE(D) this SET, and optionally, DERIVE(S) OUTPUTS or UPDATE(S) SETS, ENTITIES, GROUPS,

Syntax:

```

                                [      set-      ]
USED BY process-name(s) [ TO { DERIVE } *output- ]
                                [      {      } entity-   name(s) ] ;
                                [      { UPDATE } group-   ]
                                [      element-  ]

```

* Output-name(s) may only be used with the DERIVE clause.

Complementary Statements:

USES, UPDATES or DERIVES statement in a PROCESS section and DERIVED or UPDATED statement in SET, ENTITY, GROUP or ELEMENT sections.

Usage Rules:

-Several PROCESSES may use a SET

Synonyms:

DFV UPD

Examples:

- USED BY PROCESS-INTEGERS;
- USED BY PROC-MU-A101, PROC-MU-A102 TO DERIVE OUTPUT-1;

VOLATILITY-MEMBER statement

SET section

Purpose:

To give a measure of the changability of the contents of the SET.

Syntax:

VOLATILITY-MEMBER ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

-Only one VOLATILITY-MEMBER statement may be given for any SET.

Synonyms:

VCLM

Examples:

- VOLATILITY-MEMBER;

ALL THE ENTITIES ARE ACCESSED AT LEAST ONCE A WEEK;

AD-A060 684 MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 F19628-76-C-0197

MICHIGAN UNIV ANN ARBOR DEPT OF INDUSTRIAL AND OPERA--ETC F/G 9/2
USER REQUIREMENTS LANGUAGE (URL) USER'S MANUAL. PART II. (REFER--ETC(U)
MAR 77 F1962A-76-C-0187

ESD-TR-78-129-VOL-2 NL

NL

AD
A060684



DATE
FILMED

01-79

DDC

VOLATILITY-SET statement

SET section

Purpose:

To give a measure of the changability of the SET.

Syntax:

VOLATILITY-SET ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

-Only one VOLATILITY-SET statement may be given for any SET.

Synonyms:

VCLS

Examples:

- VOLATILITY-SET;

THIS SET WILL BE UPDATED TWICE DAILY ;

4.20 UNIT section header statement

Purpose:

To allow a detailed description of a UNIT. A UNIT is something that is used in measuring a RESOURCE. It is used in recording and estimating the resource consumption in the target system.

Syntax:

UNIT name(s) ;

Usage Rules:

- It must be the first statement in a UNIT section.
- Several UNITS may be defined at once.

Synonyms:

NONE

Examples:

- UNIT MILLI-SECOND, DOLLAR;
- UNIT MAN-HOURS;

ASSERT statement

UNIT section

Purpose:

To associate assertions about the attributes of names with other names for the purposes of consistency checking.

Syntax:

ASSERT name attribute-name attribute-value

[, name attribute-name attribute-value] ...;

Complementary Statements:

Ncne.

Usage Rules:

- Name may be any type of name.

Synonyms:

ASRT

Examples:

- ASSERT data-name-1 type character;
- ASPT sine-function arguments 1,
coord-function arguments 2;

ATTRIBUTES statement

UNIT section

Purpose:

To specify properties or characteristics particular to a given section.

Syntax:

```
ATTRIBUTES ARE attr-name { attrv-name } [ ,attr-name { attrv-name } ] ...  
                        { integer } [ { integer } ]
```

Complementary Statements:

ncne.

Usage Rules:

-A name may have several ATTRIBUTES

Synonyms:

ATTR ATTRIBUTE

Examples:

- ATTRIBUTES ARE FORMAT NUMERIC, LENGTH 6;
- ATTRIBUTES ARE FREQUENCY 100, VOLUME 10;
- ATTR CHAR ZZZ9V9;

DESCRIPTION statement

UNIT section

Purpose:

To give a text DESCRIPTION of the section being described, and to state any information which cannot be easily or accurately stated with the syntax applicable for a given section.

Syntax:

DESCRIPTION ;
comment-entry ;

Complementary Statements:

None.

Usage Rules:

- See chapter 2, section 10, for the rules concerning comment entries.

Synonyms:

DESC

Examples:

DESCRIPTION;

THIS ALLOWS YOU TO DESCRIBE IN NARRATIVE FORM WHAT YOU EXPECT THIS SECTION TO DO;

DESC;

ANY RELEVANT INFORMATION GOES HERE;

KEYWORDS statement

UNIT section

Purpose:

To selectively retrieve information from the URA data-base. A collection of information may be marked with a unique identifier (KEY) and later retrieved.

Syntax:

KEYWORDS ARE keyword-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for a keyword.

Usage Rules:

-A section may have several KEYWORDS

Synonyms:

KEY KEYWORD

Examples:

- KEYWORD IS PAYROLL;
- KEY IS CON-C1;
- KEYWORDS ARE EMP, EMPL, EMPLOYEE;

MEASURES statement

UNIT section

Purpose:

To give the RESOURCE names that the UNIT is used to measure.

Syntax:

MEASURES resource-name(s);

Complementary Statements:

MEASURED statement in RESOURCE section.

Usage Rules:

- A UNIT may measure several RESOURCES. A RESOURCE, however, may be measured only in one UNIT.

Synonyms:

MSRS

Examples:

- MEASURES CPU-TIME, REAL-TIME;
- MSRS FUNDS;

RESPONSIBLE-PROBLEM-DEFINER statement

UNIT section

Purpose:

To associate the PROBLEM-DEFINER with those sections for which he is RESPONSIBLE.

Syntax:

RESPONSIBLE-PROBLEM-DEFINER IS problem-definer-name ;

Complementary Statements:

RESPONSIBLE FOR statement in PROBLEM-DEFINER section.

Usage Rules:

- Only one PROBLEM-DEFINER may be RESPONSIBLE for any section, hence, this statement may only be used once per section.

Synonyms:

RFD

Examples:

- RESPONSIBLE-PROBLEM-DEFINER IS AL-DICKEY;
- RPD A-HERSHEY;

SECURITY statement

UNIT section

Purpose:

To associate SECURITY keys with a section which may be used to limit access to the information given in this section.

Note: The SECURITY given refers to the Problem Statement information, not the information in the target system.

Syntax:

SECURITY IS security-name(s) ;

Complementary Statements:

APPLIES statement in a DEFINE section for a SECURITY.

Usage Rules:

- A name may have several SECURITIES.

Synonyms:

SEC SECURITIES

Examples:

- SECURITY IS PROJECT-MANAGER;
- SECURITIES ARE D-OPMISTON, S-MENNEL;
- SFC L-HANNON;

SEE-MEMO statement

UNIT section

Purpose:

To indicate that information related to this section, and possibly other sections, is contained within the documentation. The information is contained in the MEMO(S) designated herein.

Syntax:

SEE-MEMO memo-name(s) ;

Complementary Statements:

APPLIES statement in a MEMO section.

Usage Rules:

- A section may have several such statements.

Synonyms:

SM SEE-MEMOS

Examples:

- SEE-MEMO BW-05-03-75-01;
- SEE-MEMOS: PROJ-MGR-106, PROJ-MGR-109;
- SM EPB-37, EPB-38;

SOURCE statement

UNIT section

Purpose:

To identify information not contained within the system documentation that is relevant to the understanding of the system. The SOURCE may be a person, a document (such as a practice or guideline), etc.

Syntax:

SOURCE IS source-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for SOURCE name.

Usage Rules:

- A name may have several SOURCES.

Synonyms:

SRC SOURCES

Examples:

- SOURCE IS ENG-LETTER-1-MAY-1973;
- SOURCE: SDP-3-0;

SYNONYMS statement

UNIT section

Purpose:

To give SYNONYMS for the name of the section. Can be used to define short forms for section-names in the documentation. Also can be used to resolve name conflicts within the system. Thus it is useful for reducing the manual effort of documentation.

Syntax:

SYNONYMS ARE synonym-name(s) ;

Complementary Statements:
DESIGNATE section.

Usage Rules:

- A name may have several SYNONYMS.

Synonyms:

SYN SYNONYM

Examples:

- SYNONYMS ARE U-11, UNIT-11;
- SYNONYM IS UNIT-11;
- SYN ALPHA;

TRACE-KEY statement

UNIT section

Purpose:

To associate a list of trace-keys with a name so that correspondences between objects in different data bases may be made.

Syntax:

TRACE-KEY trace-key-name(s) ;

Complementary Statements:

APPLIES statement in DEFINE section for TRACE-KEY name.

Usage Rules:

- The names in the name list must be trace-key names.

Synonyms:

TKEY

Examples:

- TRACE-KEY module-a;
- TKEY part-1, part-2;

APPENDIX A

Implementation Restrictions

A user defined name can have a maximum length of 30 characters (letters, digits, dashes).

The User Requirements Analyzer (URA) will ignore card columns 73 through 80 (if card input is used). Thus, only columns 1 through 72 can be used for URL statements.

Each URL input line can contain either part of a URL statement or several statements.

Any URL statement may be broken anywhere a blank is allowed.

APPENDIX B

UPL Reserved Words

A
AN
AND
APP
APPLIES
ARE
AS
ASCC
ASCD
ASET
ASSERT
ASSOCIATED
ASSOCIATED-DATA
AT
ATTR
ATTRIBUTE
ATTRIBUTES
ATTRIBUTE-VALUE
ATTV
BEC
BECG
BECOMES
BECOMING
BECS
BEING
BETWEEN
BOX
BTWN
BY
CAL
CALLED
CARD
CARDINALITY
CAUSED
CAUSES
CLASSIFICATION
CLASSIFICATIONS
CLS
CNSS
CNSD
CNTD
COND
CONDITION
CONDITIONS
CONN
CONNECTIVITY
CONSISTS
CONSUMED
CONSUMES
CONTAINED
CSD

CSS
CSTS
DEF
DEFINE
DERIVATION
DERIVED
DERIVES
DESC
DESCRIPTION
DESG
DESIGNATE
DRV
DRVD
DRVN
DRVS
ELF
ELEMENT
ELEMENTS
ENT
ENTITIES
ENTITY
EV
EVENT
EVENTS
EVT
F
FALSE
FOR
FROM
GEND
GENERATED
GENERATES
GENS
GR
GROUP
GROUPS
HAP
HAPPENS
IDD
IDENTIFIED
IDENTIFIES
IDS
IN
INCC
INCEPTION
INCEPTION-CAUSES
INCP
INP
INPUT
INPUTS
INTD
INTERFACE
INTERFACES
INTERRUPTED
INTERRUPTS
INTERVAL

INTERVALS
INTF
INTS
IS
IT
KEY
KEYWORD
KEYWORDS
MADE
MAILBOX
MAILBOXES
MAINTAINED
MAINTAINS
MAK
MAKES
MBX
MEASURED
MEASURES
MEMO
MEMOS
MSRD
MSRS
MTND
MTNS
NEGINF
OCCS
OCCURRENCES
OF
ON
ORGANIZATIONAL-UNIT
OPGU
OUT
OUTPUT
OUTPUTS
PART
PD
PFF
PERFORMED
PERFORMS
PFMD
PFMS
POSING
PRC
PRCD
PRCR
PRD
PROBLEM-DEFINER
PROBLEM-DEFINERS
PROC
PROCEDURE
PROCESS
PROCESSES
PROCESSOR
PROCESSORS
PROCR
PCVD

RCVS
REAL-WOLRD-ENTITIES
REAL-WOLRD-ENTITY
RECEIVED
RECEIVES
REL
RELATED
RELATION
RELATIONS
RES
RESOURCE
RESOURCE-USAGE
RESOURCE-USAGE-PARAMETER
RESOURCE-USAGE-PARAMETER-VALUE
RESP
RESPONSIBLE
RESPONSIBLE-INTERFACE
RESPONSIBLE-PROBLEM-DEFINER
RINT
PLN
RPD
RRWE
RSC
RU
RUP
RUPV
RUP-VALUE
RWE
SFC
SECURITIES
SECURITY
SAR
SECURITY-ACCESS-RIGHT
SECURITY-ACCESS-RIGHTS
SEE-MEMO
SEE-MEMOS
SET
SETS
SM
SOURCE
SOURCES
SRC
SSCA
SSCN
SST
SSTS
SUBP
SUBPARTS
SUBSET
SUBSETS
SUBSETTING-CRITERIA
SUBSETTING-CRITERION
SYN
SYNONYM
SYNONYMS
SYSP

SYSPAR
SYSTEM-PARAMETER
SYSTEM-PARAMETERS
T
TERC
TERM
TERMINATED
TERMINATES
TERMINATION
TERMINATION-CAUSES
THE
THIS
THRU
THRU
TIMES-PER
TIMP
TKEY
TO
TRACE-KEY
TRGD
TRGS
TRIGGERED
TRIGGERS
TRMD
TRMS
TRUE
UNIT
UPDATE
UPDATED
UPD
UPDATES
UPDD
UPDS
USED
USES
USG
USING
UTLD
UTLS
UTILIZED
UTILIZES
VAL
VALUE
VALUES
VIA
VOL
VOLATILITY
VOLATILITY-MEMBER
VOLATILITY-SET
VOLM
VOLS
WHEN
WHETHER
WHILE
WHL
WITH

APPENDIX C

URL Optional Words

A
AN
AND
ARE
AS
AT
BEING
BY
FOR
FROM
IN
IS
IT
OF
ON
THE
THIS
TO
WHETHER
WITH

APPENDIX D

Reserved Words with Synonyms

APPLIES	APP
ASSERT	ASRT
ASSOCIATED	ASOC
ASSOCIATED-DATA	ASOD
ATTRIBUTE	ATTRIBUTES ATTR
ATTRIBUTE-VALUE	ATTV
BECOMES	BECS
BECOMING	BEC BECG
BETWEEN	BTWN
CALLED	CAL
CARDINALITY	CARD OCCS OCCURRENCES
CAUSED	CSD
CAUSES	CSS
CLASSIFICATION	CLASSIFICATIONS CLS
CONDITION	COND CONDITIONS
CONNECTIVITY	CONN
CONSISTS	CSTS
CONSUMED	CNSD
CONSUMES	CNSS
CONTAINED	CNTD
DEPINE	DEF
DERIVATION	DRVN
DERIVE	DRV
DERIVED	DRVD
DERIVES	DRVS
DESCRIPTION	DESC
DESIGNATE	DESG
ELEMENT	ELE ELEMENTS
ENTITY	ENT ENTITIES
EVENT	EV EVT EVENTS
FALSE	F
GENERATED	GEND
GENERATES	GENS
GROUP	GR GROUPS
HAPPENS	HAP
IDENTIFIED	IDD
IDENTIFIES	IDS
INCEPTION	INCP
INCEPTION-CAUSES	INCC
INPUT	INP INPUTS
INTERFACE	INTF INTERFACES
	ORGANIZATIONAL-UNIT ORGU
	RWE REAL-WORLD-ENTITY
INTERRUPTED	INTD
INTERRUPTS	INTS
INTERVAL	INT INTERVALS
KEYWORD	KEY KEYWORDS
MADE	
MAILBOX	BOX MBX MAILBOXES
MAINTAINED	MTND
MAINTAINS	MTNS
MAKES	MAK
MEASURED	MSRD
MEASURES	MSRS
MEMO	MEMOS

APPENDIX D

Reserved Words with Synonyms

NEGINF	
OUTPUT	OUT OUTPUTS
PART	
PER	
PERFORMED	PFMD
PERFORMS	PFMS
POSINF	
PROBLEM-DEFINER	PD PROBLEM-DEFINERS
PROCEDURE	PRCD PRD
PROCESS	PROC PRC PROCESSES
PROCESSOR	PRCR PROCR PROCESSORS
RECEIVED	RCVD
RECEIVES	RCVS
RELATED	REL
RELATION	RLN RELATIONS
RESOURCE	RSC
RESOURCE-USAGE	RU
RESOURCE-USAGE-PARAMETER	RUP
RESOURCE-USAGE-PARAMETER-VALUE	RUPV RUP-VALUE
RESPONSIBLE	RESP RES
RESPONSIBLE-INTERFACE	RINT
RESPONSIBLE-PROBLEM-DEFINER	RPD
SECURITY	SEC SECURITIES
SECURITY-ACCESS-RIGHT	SAR
	SECURITY-ACCESS-RIGHTS
SEE-MEMO	SM SEE-MEMOS
SET	SETS
SOURCE	SRC SOURCES
SUBPARTS	SUBP
SUBSET	SST
SUBSETS	SSTS
SUBSETTING-CRITERIA	SSCA
SUBSETTING-CRITERION	SSCN
SYNONYM	SYN SYNONYMS
SYSTEM-PARAMETER	SYSP SYSPAR
	SYSTEM-PARAMETERS
TERMINATED	TRMD
TERMINATES	TRMS
TERMINATION	TERM
TERMINATION-CAUSES	TERC
TIMES-PER	TIMP
TRACE-KEY	TKEY
TRIGGERED	TRGD
TRIGGERS	TRGS
TRUE	T
UNIT	
UPDATE	UPD
UPDATED	UPDD
UPDATES	UPDS
USED	
USES	
USING	USG
UTILIZED	UTLD
UTILIZES	UTLS
VALUES	VAL VALUE

APPENDIX D

Reserved Words with Synonyms

VOLATILITYVOL
VOLATILITY-MEMBERVOLM
VOLATILITY-SETVOLS
WHILEWHL

APPENDIX E

Name Types

ATTRIBUTE
ATTRIBUTE-VALUE
CLASSIFICATION
CONDITION
ELEMENT
ENTITY
EVENT
GROUP
INPUT
INTERFACE
INTERVAL
KEYWORD
MAILBOX
MEMO
OUTPUT
PROBLEM-DEFINER
PROCESS
PROCESSOR
RELATION
RESOURCE
RESOURCE-USAGE-PARAMETER
SECURITY
SOURCE
SET
SUPSETTING-CRITERION
SYSTEM-PARAMETER
TRACE-KEY
UNIT

APPENDIX F

Section Types

CONDITION
DEFINE
DESIGNATE
ELEMENT
ENTITY
EVENT
GROUP
INPUT
INTERFACE
INTERVAL
MEMO
OUTPUT
PROBLEM-DEFINER
PROCESS
PROCESSOR
RELATION
RESOURCE
RESOURCE-USAGE-PARAMETER
SET
UNIT

APPENDIX G

URL Forms

The following hard-copy forms are intended to aid the user in writing URL according to the specifications given in the URL Reference Manual. The forms for a section give all statements allowed in that section and thus help the user to keep all possibilities in mind while writing his requirements. They also simplify the keypunching process.

CODING INSTRUCTIONS

The following general comments apply to the forms for all section types:

1. All statements are optional; the user should make use of only those he requires.
2. A continuation form is furnished for those statements which are too long for the space provided. To use this, the problem-definer should first state the section type and name at the top of the page, then, below, express the continuations as complete statements. (The abbreviations from Appendix D of the URL Reference Manual may be used for statement names.) A name-list should be broken only at the end of a name.

DESIGNATE statements, of the form:

DESIGNATE name AS A synonym FOR name [, name AS A SYNONYM FOR name]...;

should be entered on continuation forms.

KEYPUNCHING INSTRUCTIONS

A statement should be keypunched only if it contains material coded by the user. For most statements, one may recognize the end of the statement by the semi-colon which is to be punched after it. The only exceptions to this rule are the comment-entry statements (DESCRIPTION, TRUE-WHILE, FALSE-WHILE, VOLATILITY, VOLATILITY-SET, VOLATILITY-MEMBER, DERIVATION, and PROCEDURE) which have two parts, each followed by a semi-colon. The first part consists of the printed statement name, while the second part contains only user-defined material. Both parts of a comment-entry statement should be keypunched if any coding appears in the second part of the statement. Otherwise, neither part of the statement should be punched.

Form titles, system name, dates and page numbers are not to be keypunched.

Columns 73-80 of each card will be ignored and therefore should not be used for URL statements. A URL statement may be punched on more than one card, and may be broken anywhere a blank is allowed.

URL DEFINITION FORM

400

----- PAGE ____ OF ____
system name ----- date -----

DEFINE -----;
(name)

☐ ATTRIBUTE;

☐ SECURITY;

☐ ATTRIBUTE-VALUE;

☐ SOURCE;

☐ CLASSIFICATION;

☐ SUBSETTING-CRITERION;

☐ KEYWORD;

☐ SYSTEM-PARAMETER;

☐ MAILBOX;

☐ TRACE-KEY;

APPLIES TO -----;
(list of appropriate names)
(only for keyword, mailbox, security, source and trace-key)

ASSET -----;
(list of names followed by attribute-names
and attribute values)

ATTRIBUTES ARE -----,
(attribute name) (attribute value)

DESCRIPTION;

-----;
(narrative description)

KEYWORDS -----;
(list of keywords)

THRU ----- ;
 (maximum value)
 (only for system parameter)
 (may be used only if the VALUE statement is not used)

402

(narrative description)

system name date

PAGE --- OF ---

MAINTAINED BY -----;
(list of process names)
(only for subsetting criterion)

RESPONSIBLE-PROBLEM-DEFINER _____;
(name of responsible problem definer)

SECURITY -----: (list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----: (list of sources of information)

SUBSETTING-CRITERION FOR _____;
(list of set names)
(only for subsetting-criterion)

SYNONYMS -----: (list of synonyms)

TRACE-KEY -----
(list of trace-key names)

VALUE -----: (value)
(only for system parameter)

VALUES -----
 (minimum value)
 (only for system parameter)
 (may be used only if the VALUE statement is not used)

THRU ----- (maximum value)
(only for system parameter)
(may be used only if the VALUE statement is not used)

----- PAGE ____ OF -----

----- system name ----- date -----

KEYWORDS -----
(list of keywords)

MADE TRUE BY _____
(list of event, input, and process names)

MADE FALSE BY _____
(list of event, input, and process names)

RESPONSIBLE-PROBLEM-DEFINER _____
(name of responsible problem definer)

SECURITY -----
(list of applicable security names)

SEE MEMO _____
(list of memo names)

SOURCE -----
(list of sources of information)

SYNONYMS -----
(list of synonyms)

TRACE-KEY -----
(list of trace-key names)

TRUE WHILE;

(comment-entry)

FALSE WHILE;

(comment entry)

404

----- PAGE ----- OF -----

date

```

ASSERT -----;
          (list of names followed by attribute names
           and attribute values)

```

ATTRIBUTES ARE

(attribute name)	(attribute value)
------------------	-------------------

CONTAINED IN _____
(list of group, entity, input and output names)

DEIVED BY -----
(list of process names)

USING (list of input, entity, set, group and element names);

DERIVED BY _____
(list of process names)

(narrative description)

IDENTIFIES -----
(list of entity names)

KEYWORDS -----
(list of keywords)

RESPONSIBLE-PROBLEM-DEFINER _____
(name of responsible problem definer)

SECURITY -----
(list of applicable security names)

----- PAGE ____ OF -----
system name ----- date -----

SEE-MEMO -----
(list of memo names)

SOURCE -----
(list of sources of information)

SUBSETTING-CRITERION FOR -----
(list of set names)

SYNONYMS -----
(list of synonyms)

TRACE-KEY -----
(list of trace-key names)

UPDATED BY -----
(list of process names)

USING -----
(list of input, set, entity, group and element names)

UPDATED BY -----
(list of process names)

USED BY -----
(list of process names)

TO DERIVE -----
(list of set, entity, group, element and
output names)

USED BY -----
(list of process names)

TO UPDATE -----
(list of set, entity, group and element names)

USED BY -----
(list of process names)

VALUE -----
(value)

VALUES ----- THEN -----
(minimum value) (maximum value)
(may be used only if the VALUE statement is not used)

406

VIA _____
(relation name)

system name date

PAGE ____ OF ____

RESPONSIBLE-PROBLEM-DEFINER _____;
(name of responsible problem definer)

SECURITY -----; (list of applicable security names)

SEE-MEMO -----
(list of memo names)

SOURCE -----; (list of sources of information)

SYNONYMS -----;
 (list of synonyms)

TRACE-KEY -----; (list of trace-key names)

UPDATED BY _____
(list of process names)

```

USING (list of input, set, entity, group, or element names);

```

UPDATED BY -----
(list of process names)

USED BY -----
(list of process names)

TO DERIVE (list of set, entity, group, element, and output names);

USED BY _____;
(list of process names)

TO UPDATE -----;
(list of set, entity, group and element names)

USED BY -----
(list of process names)

VOLATILITY;

(comment-entry: changeability of the entity)

408

MAKES _____ TRUE;
(list of condition names)

system name date PAGE OF -----

```

MAKES ----- FALSE;
              (list of condition names)

```

RESPONSIBLE-PROBLEM-DEFINER _____;
(name of responsible problem definer)

SECURITY -----; (list of applicable security names)

SEE MEMO -----; (list of memo names)

SOURCE -----: (list of sources of information)

SYNONYMS -----; (list of synonyms)

```

TERMINATES -----;
                (list of process names)

```

ON TERMINATION OF _____;
(list of process names)

TRACE-KEY -----;

(list of trace-key names)

TRIGGERS -----: (list of process names)

410

----- system name	----- date
----------------------	---------------

KEYWORDS _____
(list of keywords)

----- PAGE ____ OF -----
system name ----- date -----
RESPONSIBLE-PROBLEM-DEFINER -----;
(name of responsible problem definer)
SECURITY -----;
(list of applicable security names)
SEE-MEMO -----;
(list of memo names)
SOURCE -----;
(list of sources of information)
SUBSETTING-CRITERION FOR -----;
(list of set names)
SYNONYMS -----;
(list of synonyms)
TRACE-KEY -----;
(list of trace-key names)
UPDATED BY -----;
(list of process names)
USING -----;
(list of input, set, entity, group or element names)
UPDATED BY -----;
(list of process names)
USED BY -----;
(list of process names)
TO DEFINE -----;
(list of set, entity, group, element and
output names)
USED BY -----;
(list of process names)
TO UPDATE -----;
(list of set, entity, group and element names)
USED BY -----;
(list of process names)

URL INPUT DEFINITION FORM

412

----- PAGE ____ OF -----

system name ----- date -----

INPUT -----;
(name of input)

ASSEPT -----;
(list of names followed by attribute names
and attribute-values)

ATTRIBUTES ARE -----;
(attribute name) (attribute value)

-----;
-----;

CAUSES -----;
(list of event names)

CLASSIFICATION -----;
(list of classification names
optionally followed by classification levels)

CONSISTS OF -----;
(list of group and element names,
optionally preceded by system-parameters)

CONTAINED IN -----;
(list of set names)

DESCRIPTION;
-----;
-----;
-----;
(narrative description)

GENERATED BY -----;
(list of interface names)

HAPPENS -----;
(system-parameter)

TIMES PER -----;
(interval name)

INTERRUPTS -----;
(list of process names)

KEYWORDS -----;
(list of keywords)


```

-----
system name      date      PAGE ____ OF ____
MAKES  ----- TRUE;
              (list of condition names)
MAKES  ----- FALSE;
              (list of condition names)
PART OF -----;
              (name of input)
RECEIVED BY -----;
              (list of process names)
RESPONSIBLE-PROBLEM-DEFINER -----;
              (name of responsible problem definer)
SECURITY -----;
              (list of applicable security names)
SEE-MEMO -----;
              (list of memo names)
SOURCE -----;
              (list of sources of information)
SUBPAPTS ARE -----;
              (list of input names)
SYNONYMS -----;
              (list of synonyms)
TERMINATES -----;
              (list of process names)
TRACE-KEY -----;
              (list of trace-key names)
TRIGGERS -----;
              (list of process names)
USED BY -----;
              (list of process names)
      TO DERIVE -----;
              (list of set, entity, group, element, and output
              names)
USED BY -----;
              (list of process names)
      TO UPDATE -----;
              (list of set, entity, group, and element names)
USED BY -----;
              (list of process names)

```

414

PAGE _____ OF _____

```
INTERFACE -----;
```

ASSEPT -----;
(list of names followed by attribute-names
and attribute-values)

ATTRIBUTES ARE	(attribute name)	(attribute value)

DESCRIPTION:

(narrative, description)

```

GENERATES -----;
              (list of input names)

```

KEYWORDS -----: (list of keywords)

PART OF -----;
(interface name)

system name-----
date

PAGE ____ OF ____

RECEIVES -----;
(list of output names)RESPONSIBLE FOR -----;
(list of set names)RESPONSIBLE-PROBLEM-DEFINER -----;
(name of responsible problem definer)SECURITY -----;
(list of applicable security names)SECURITY-ACCESS-RIGHT -----;
(list of classification names
optionally followed by
classification levels)SEE-MEMO -----;
(list of memo names)SOURCE -----;
(list of sources of information)SUBPARTS ARE -----;
(list of interface names)SYNONYMS -----;
(list of synonyms)TRACE-KEY -----;
(list of trace-key names)

416

PAGE OF

INTERVAL -----;
(name of interval)

```

ASSERT -----;
      (list of names followed by attribute-names
       and attribute-values)

```

[illegible]

CONSISTS OF -----:
(list of interval names, optionally preceded by
system parameters)

DESCRIPTION:

(narrative description)

KEYWORDS -----
(list of keywords)

RESPONSIBLE PROBLEM DEFINER _____
(name of responsible problem definer)

SECURITY -----
(list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----
(list of sources of information)

SYNONYMS -----
(list of synonyms)

TRACE-KEY -----
(list of trace key names)

417

TRACE-KEY -----;
(list of trace key names)

URL OUTPUT DEFINITION FORM

418

system name . date PAGE ____ OF ____

OUTPUT -----;
(name of output)

ASSERT -----;
(list of names followed by attribute-names
and attribute-values)

ATTRIBUTES ARE -----'
(attribute name) (attribute value)
-----'
-----';

CLASSIFICATION -----;
(list of classification names
optionally followed by classification levels)

CONSISTS OF -----;
(list of group and element names,
optionally preceded by system-parameters)

CONTAINED IN -----;
(list of set names)

DERIVED BY -----
(list of process names)

USING -----;
(list of input, set, entity, group and element names)

DERIVED BY -----;
(list of process names)

DESCRIPTION;

-----;
(narrative description)

GENERATED BY -----;
(list of process names)

HAPPENS -----
(system-parameter)

TIMES-PER -----;
(interval name)

system name date

PAGE OF

KEYWORDS ----- (list of keywords) -----;

PART OF _____;
(name of output)

RECEIVED BY _____;
(list of interface names)

RESPONSIBLE PROBLEM DEFINER _____;
(name of responsible problem definer)

SECURITY -----; (list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----: (list of sources of information)

SUBPARTS ARE -----;
 (list of output names)

SYNONYMS ----- (list of synonyms) ----- :

TRACE-KEY -----; (list of trace-key names)

420

TRACE KEY -----
(list of trace-key names)

421

```

INTERRUPTS -----:
                (list of process names)

```

----- PAGE ____ OF ____
system name date

KEYWORDS -----;
(list of keywords)

MAINTAINS -----;
(list of relation or subsetting-criteria names)

MAKES ----- TRUE;
(list of condition names)

MAKES ----- FALSE;
(list of condition names)

PART OF -----;
(process name)

PERFORMED BY -----;
(name of processor)

PROCEDURE;

-----;
(comment entry: description of procedure)

RECEIVES -----;
(list of input names)

RESOURCE USAGE ----- FOR -----;
(system-parameter) (name of resource-
usage-parameter)

RESPONSIBLE-PROBLEM-DEFINER -----;
(name of responsible problem definer)

```

----- PAGE ____ OF -----
system name ----- date -----

SECURITY -----
              (list of applicable security names)

SECURITY-ACCESS-RIGHT -----
              (list of classification names
              optionally followed by
              classification levels)

SEE-MEMO -----
              (list of memo names)

SOURCE -----
              (list of sources of information)

SUBPARTS ARE -----
              (list of process names)

SYNONYMS -----
              (list of synonyms)

TERMINATED BY -----
              (list of event, input, and process names)

TERMINATED WHEN ----- BECOMES TRUE;
              (name of condition)

TERMINATED WHEN ----- BECOMES FALSE;
              (name of condition)

TERMINATES -----
              (list process names)

TERMINATION-CAUSES -----
              (list of event names)

TRACE-KEY -----
              (list of trace-key names)

TRIGGERED BY -----
              (list of event, input, and process names)

TRIGGERED WHEN ----- BECOMES TRUE;
              (name of condition)

TRIGGERED WHEN ----- BECOMES FALSE;
              (name of condition)

TRIGGERS -----
              (list of process names)

```

PAGE OF

UPDATES -----
(list of entity, set, group, and element names)

```

USING _____;
      (list of input, set, entity, group, and element names)

```

```

UPDATES -----;
              (list of entity, set, group, and element names)

```

USES -----
(list of set, group, element, input, and entity names)

TO DERIVE -----;
(list of set, entity, group, element,
and output names)

USES -----
(list of set, group, element, input, and entity names)

TO UPDATE -----;
(list of set, entity, group, and element names)

USES ----- (list of set, group, element, input, and entity names);

UTILIZED BY _____;
(list of process names)

```
UTILIZES ----- ;
              (list of process names)
```

system name-----
date

PAGE ____ OF ____

PROCESSOR -----;
(name of processor)ASSERT -----;
(list of names followed by attribute names
and attribute values)ATTRIBUTES ARE -----
(attribute name) (attribute value)-----
-----;CONSUMES ----- AT RATE OF
(name of resource)----- PPF -----;
(system-parameter) (name of
resource-usage-parameter)

DESCRIPTION;

-----;
(narrative description)

TRACE-KEY _____
(list of trace key names)

427

SYSTEM NAME

date

PAGE ____ OF ____

RELATION

(name of relation)

ASSERT

```
(list of names followed by attribute-names
      and attribute-values)
```

ASSOCIATED DATA IS

```
-----
(list of element and group names)
```

ATTRIBUTES ARE

(attribute name)

(attribute value)

BETWEEN

(name of entity)

AND

(name of entity)

CARDINALITY IS

(system-parameter)

CONNECTIVITY IS

```

-----
(system-parameter)

```

TO

```
-----
(system parameter)
```

DERIVATION:

(derivation rules)

----- PAGE ____ OF -----

system name date

DESCRIPTION:

(narrative description)

KEYWORDS -----
(list of keywords)

MAINTAINED BY _____
(list of process names)

RESPONSIBLE PROBLEM-DEFINER (name of responsible problem definer)

SECURITY -----
(list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----
(list of sources or information)

SYNONYMS -----
(list of synonyms)

TRACE-KEY -----
(list of trace-key names)

429

PAGE ____ OF ____

system name _____ date _____

RESOURCE _____;
(name of resource)

ASSERT _____;
(list of names followed by attribute-names
and attribute-values)

ATTRIBUTES ARE _____;
(attribute name) (attribute value)

CONSUMED BY _____ AT RATE OF _____;
(list of processor names)

_____ PER _____;
(system-parameter) (name of
resource-usage-parameter)

DESCRIPTION;

(narrative description)

KEYWORDS _____;
(list of keywords)

MEASURED IN _____;
(name of unit)

RESPONSIBLE PROBLEM-DEFINER _____;
(name of responsible problem definer)

SECURITY _____;
(list of applicable security names)

SEE-MEMO _____;
(list of memo names)

SOURCE _____;
(list of sources of information)

SYNONYMS _____;
(list of synonyms)

TRACE KEY _____;
(list of trace-key names)

----- PAGE ____ OF -----
 system name date

```
RESOURCE-USAGE-PARAMETER -----;
                        (name of
                        resource-usage-parameter)
```

ASSET -----
(list of names followed by attribute-names
and attribute-values)

ATTRIBUTES ARE

(attribute name)	(attribute value)

DESCRIPTION:

(narrative description)

KEYWORDS -----
(list of keywords)

RESOURCE-USAGE-PARAMETER-VALUE -----
(system-parameter)

FOR _____
(name of process)

RESPONSIBLE PROBLEM-DEFINER (name of responsible problem definer)

SECURITY -----
(list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----
(list of sources of information)

SYNONYMS -----
(list of synonyms)

TRACE-KEY -----
(list of trace-key names)

431

-----	-----
system name	date

```

ASSERT -----:
      (list of names followed by attribute-names
       and attribute-values)

```

CARDINALITY IS -----
(system-parameter)

CLASSIFICATION -----
(list of classification names
optionally followed by classification levels)

CONSISTS OF _____
(list of entity, input, and output names,
optionally preceded by system-parameters)

(comment entry: derivation rules)

DERIVED BY _____
(list of process names)

USING (list of input, set, entity, group and element names)

DERIVED BY _____
(list of process names)

(narrative description)

system name

date

PAGE 1 OF 1

KEYWORDS -----
(list of keywords)

RESPONSIBLE INTERFACE -----
(list of interface names)RESPONSIBLE-PROBLEM-DEFINER _____;
(name of responsible problem definer)

SECURITY -----
(list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----
(list of sources of information)

SUBSET OF -----
(list of set names)SUBSETS ARE -----
(list of set names)

SUBSETTING CRITERIA ARE

system name

date

PAGE ____ OF ____

UPDATED BY _____
(list of process names)

```

USING (list of input, set, entity, group, and element names);

```

UPDATED BY _____;
(list of process names)

USED BY -----
(list of process names)

TO DERIVE -----;
(list of set, entity, group, element,
and output names)

USED BY -----
(list of process names)

TO UPDATE -----; (list of set, entity, group, and element names)

USED BY -----;
(list of process names)

VOLATILITY-MEMBER:

(comment-entry: changeability of a member of the set)

VOLATILITY SET;

(comment-entry: changeability of 'the set')

434

[illegible]

```

ASSERT -----
      (list of names followed by attribute-names
       and attribute values)

```

DESCRIPTION:

(narrative description)

KEYWORDS -----
(list of keywords)

MEASURES -----
(list of resource names)

RESPONSIBLE-PROBLEM-DEFINER (name of responsible problem definer)

SECURITY -----
(list of applicable security names)

SEE MEMO -----
(list of memo names)

SOURCE -----
(list of sources of information)

SYNONYMS -----
(list of synonyms)

TRACE-KEY _____
(list of trace-key names)

URL CONTINUATION FORM

435 :

system name

date

PAGE ____ OF ____

----- (section type) ----- (name) -----

[illegible]

INDEX

Computer 2

data-base 53, 68, 88, 109, 129, 150, 171, 190, 207, 219, 235, 249,
267, 297, 317, 330, 342, 359, 378

language 2

APPLIES 10, 26, 53, 56, 57, 58, 64, 68, 71, 72, 73, 88, 90, 91, 92,
109, 112, 113, 114, 129, 132, 133, 134, 150, 152, 153, 154, 171,
176, 177, 178, 190, 195, 197, 198, 207, 209, 210, 211, 215, 219,
221, 222, 235, 239, 240, 241, 249, 250, 252, 253, 254, 267, 276,
278, 279, 297, 301, 303, 304, 317, 320, 321, 322, 330, 333, 334,
335, 342, 345, 346, 347, 359, 362, 363, 364, 367, 378, 381, 382,
383, 387, 393

ASSET 6, 8, 10, 13, 15, 17, 19, 21, 23, 25, 26, 27, 29, 30, 34, 36,
39, 40, 41, 43, 46, 65, 80, 100, 121, 141, 161, 186, 203, 216,
226, 246, 258, 293, 309, 326, 339, 351, 375, 387, 393

ASSOCIATED 13, 19, 81, 142, 310, 387, 393

ASSOCIATED-DATA 36, 81, 142, 310, 387, 393

ATTRIBUTE 6, 8, 10, 13, 15, 17, 19, 21, 23, 25, 26, 27, 29, 30, 34,
36, 39, 40, 41, 43, 47, 62, 66, 82, 101, 122, 143, 162, 187,
204, 217, 227, 247, 259, 294, 311, 327, 340, 352, 376, 387, 393,
396

ATTRIBUTE-VALUE 10, 62, 387, 393, 396

BECOMES 17, 30, 32, 123, 265, 282, 286, 387, 393

BECOMING 8, 48, 49, 50, 51, 123, 265, 282, 286, 387, 393

BETWEEN 36, 110, 312, 387, 393

CALLED 387, 393

CARDINALITY 15, 36, 41, 102, 313, 315, 353, 387, 389, 393

CAUSED 17, 123, 387, 393

CAUSES 8, 17, 21, 48, 123, 124, 163, 387, 388, 393

CLASSIFICATION 10, 13, 15, 19, 21, 27, 41, 62, 83, 103, 144, 164,
228, 354, 387, 393, 396

CONDITION 8, 9, 45, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
59, 61, 120, 123, 130, 172, 265, 269, 282, 286, 387, 393, 396,
397

CONNECTIVITY 36, 314, 387, 393

CONSISTS 15, 19, 21, 25, 27, 41, 84, 104, 105, 145, 146, 165, 166,
205, 229, 230, 355, 387, 388, 393

CONSUMED 39, 295, 328, 387, 393

CONSUMES 34, 295, 387, 393

CONTAINED 13, 15, 19, 21, 27, 84, 104, 105, 140, 145, 146, 165, 166,
229, 230, 355, 387, 393

DEFINE 6, 7, 10, 11, 53, 56, 58, 62, 63, 64, 66, 67, 68, 69, 70, 71,
72, 73, 74, 75, 77, 88, 90, 92, 109, 112, 114, 115, 129, 132,
134, 150, 152, 154, 171, 176, 178, 190, 195, 198, 207, 209, 211,
212, 219, 221, 222, 223, 235, 239, 241, 243, 249, 250, 252, 254,
255, 267, 268, 276, 279, 297, 301, 304, 317, 320, 322, 330, 333,
335, 342, 345, 347, 359, 362, 364, 367, 368, 378, 381, 383, 388,
393, 397

DERIVATION 36, 41, 315, 356, 357, 388, 393, 398

DERIVE 14, 16, 20, 22, 33, 42, 97, 118, 159, 184, 260, 289, 357, 371,
388, 393

DERIVED 13, 15, 19, 27, 41, 85, 97, 106, 118, 147, 159, 184, 231,
260, 289, 315, 356, 357, 371, 388, 393

DERIVES 30, 85, 97, 106, 118, 147, 159, 184, 231, 260, 289, 357, 371,
388, 393

DESCRIPTION 6, 8, 10, 13, 15, 17, 19, 21, 23, 25, 26, 27, 29, 30, 34,
36, 39, 40, 41, 43, 52, 67, 86, 107, 125, 148, 167, 188, 206,

INDEX

218, 232, 248, 261, 296, 316, 329, 341, 358, 377, 388, 393, 398
DESIGNATE 7, 12, 59, 75, 78, 94, 115, 135, 156, 180, 200, 212, 223,
243, 255, 281, 306, 323, 336, 348, 368, 384, 388, 393, 397, 398
ELEMENT 13, 14, 79, 81, 82, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93,
94, 96, 97, 98, 104, 106, 108, 117, 118, 140, 145, 147, 149,
155, 158, 159, 165, 184, 229, 231, 260, 288, 289, 310, 357, 367,
370, 371, 388, 393, 396, 397
ENTITY 15, 16, 81, 84, 85, 87, 96, 97, 99, 101, 102, 104, 105, 106,
107, 108, 109, 110, 111, 112, 113, 114, 115, 117, 118, 119, 140,
142, 145, 146, 147, 149, 158, 159, 184, 231, 260, 288, 289, 308,
310, 312, 314, 318, 350, 355, 357, 367, 370, 371, 372, 388, 393,
396, 397
EVENT 17, 18, 45, 48, 54, 120, 122, 123, 125, 126, 127, 129, 130,
131, 132, 133, 134, 135, 137, 139, 264, 284, 286, 388, 393, 396,
397
Files 356
FALSE 8, 9, 17, 22, 30, 31, 32, 45, 48, 49, 50, 51, 54, 61, 120, 123,
130, 172, 265, 269, 282, 286, 388, 393
FREQUENCY 47, 66, 82, 101, 122, 143, 162, 187, 204, 217, 227, 247,
259, 294, 311, 327, 340, 352, 376
GENERATED 21, 27, 168, 189, 233, 262, 388, 393
GENERATES 23, 30, 168, 189, 233, 262, 388, 393
GROUP 19, 20, 84, 85, 96, 97, 104, 106, 108, 117, 118, 140, 142, 143,
145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 158,
159, 165, 184, 229, 231, 260, 288, 289, 310, 357, 367, 370, 371,
388, 393, 396, 397
HAPPENS 17, 21, 27, 30, 126, 169, 234, 263, 388, 393
IDENTIFIED 15, 87, 108, 149, 388, 393
IDENTIFIES 13, 19, 87, 108, 149, 388, 393
INCEPTION 17, 127, 264, 388, 393
INCEPTION-CAUSES 30, 127, 264, 388, 393
INPUT 21, 22, 84, 85, 96, 106, 117, 120, 140, 146, 147, 158, 160,
162, 165, 166, 167, 168, 169, 171, 172, 173, 174, 175, 176, 177,
178, 179, 180, 183, 184, 189, 205, 231, 257, 273, 286, 289, 355,
357, 370, 388, 393, 396, 397
INTERFACE 23, 24, 168, 185, 187, 188, 189, 190, 191, 192, 193, 194,
195, 196, 197, 198, 199, 200, 237, 361, 388, 389, 390, 393, 396,
397
INTERRUPTED 30, 265, 388, 393
INTERUPTS 8, 17, 21, 30, 49, 128, 170, 265, 266, 388, 389, 393
INTERVAL 25, 126, 169, 202, 204, 205, 206, 207, 208, 209, 210, 211,
212, 234, 263, 388, 389, 393, 396, 397
KEYWORD 2, 6, 8, 10, 13, 15, 17, 20, 21, 23, 25, 26, 27, 29, 31, 34,
36, 39, 40, 41, 43, 53, 62, 64, 68, 88, 109, 129, 150, 171, 190,
207, 219, 235, 249, 267, 297, 317, 330, 342, 359, 378, 389, 393,
396
MAILBOX 10, 29, 62, 64, 250, 251, 387, 389, 393, 396
MAINTAINED 10, 36, 69, 268, 318, 389, 393
MAINTAINS 31, 69, 268, 318, 389, 393
MAKE 17, 22, 31, 130, 172, 269, 389, 393
MEASURED 39, 331, 389, 393
MEASURES 43, 379, 389, 393
MEMO 6, 26, 57, 72, 91, 113, 115, 133, 153, 177, 197, 210, 212, 214,
215, 217, 218, 219, 220, 221, 222, 223, 240, 243, 253, 255, 278,
303, 321, 334, 346, 363, 368, 382, 389, 393, 396, 397
OUTPUT 27, 28, 61, 84, 97, 118, 140, 146, 159, 184, 192, 225, 227,

INDEX

229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241,
242, 243, 257, 260, 262, 355, 371, 389, 394, 396, 397

PERFORMED 31, 271, 389, 394

PERFORMS 34, 299, 389, 394

PROBLEM-DEFINER 6, 29, 55, 70, 89, 111, 131, 151, 175, 194, 208, 220,
238, 245, 247, 248, 249, 250, 251, 252, 253, 254, 255, 274, 300,
319, 332, 344, 350, 360, 380, 389, 394, 396, 397

PROCEDURE 31, 272, 389, 394, 398

PROCESS 30, 31, 32, 33, 45, 49, 50, 51, 69, 85, 96, 97, 106, 117,
118, 120, 127, 128, 137, 139, 147, 158, 159, 170, 174, 183, 184,
231, 233, 257, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268,
269, 270, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282,
284, 286, 287, 288, 289, 290, 291, 299, 318, 343, 357, 370, 371,
389, 394, 396, 397

PROCESSOR 34, 35, 292, 294, 296, 297, 298, 299, 300, 301, 302, 303,
304, 305, 306, 389, 394, 396, 397

PSA 1

PSL 1

RECEIVED 22, 29, 174, 192, 237, 273, 389, 390, 394

RECEIVES 23, 31, 174, 192, 237, 273, 390, 394

RELATED 16, 110, 312, 390, 394

RELATION 36, 38, 81, 110, 142, 268, 308, 310, 311, 312, 313, 314,
315, 316, 317, 318, 319, 320, 321, 322, 323, 390, 394, 396, 397

RESOURCE 39, 325, 327, 329, 330, 332, 333, 334, 335, 336, 374, 390,
394, 396, 397

RESOURCE-USAGE 31, 275, 390, 394

RESOURCE-USAGE-PARAMETER 40, 338, 339, 340, 341, 342, 343, 344, 345,
346, 347, 348, 349, 390, 394, 396, 397

RESOURCE-USAGE-PARAMETER-VALUE 40, 343, 390, 394

RESPONSIBLE 23, 29, 55, 70, 89, 111, 131, 151, 175, 193, 194, 208,
220, 238, 251, 274, 300, 319, 332, 344, 360, 361, 380, 390, 394

RESPONSIBLE-INTERFACE 41, 193, 361, 390, 394

RESPONSIBLE-PROBLEM-DEFINER 6, 8, 10, 14, 16, 18, 20, 22, 23, 25, 26,
28, 31, 34, 36, 39, 40, 42, 43, 55, 70, 89, 111, 131, 151, 175,
194, 208, 220, 238, 251, 274, 300, 319, 332, 344, 360, 380, 390,
394

SECURITY 6, 7, 8, 10, 14, 16, 18, 20, 22, 23, 25, 26, 28, 29, 31, 34,
36, 39, 40, 42, 43, 56, 62, 64, 71, 90, 112, 132, 152, 176, 195,
209, 221, 239, 252, 276, 301, 320, 333, 345, 362, 381, 390, 394,
396

SECURITY-ACCESS-RIGHT 23, 31, 34, 196, 277, 302, 390, 394

SEE-MEMO 6, 9, 10, 14, 16, 18, 20, 22, 23, 25, 28, 29, 31, 34, 36,
39, 40, 42, 43, 57, 72, 91, 113, 133, 153, 177, 197, 210, 215,
240, 253, 278, 303, 321, 334, 346, 363, 382, 390, 394

SET 41, 42, 74, 85, 93, 96, 97, 102, 105, 106, 117, 118, 147, 155,
158, 159, 166, 184, 193, 230, 231, 260, 288, 289, 350, 352, 353,
355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367,
368, 370, 371, 372, 373, 390, 394, 396, 397

SOURCE 6, 9, 10, 14, 16, 18, 20, 22, 23, 25, 26, 28, 29, 31, 34, 37,
39, 40, 42, 43, 58, 62, 64, 73, 92, 114, 134, 154, 178, 198,
211, 222, 241, 254, 279, 304, 322, 335, 347, 364, 383, 390, 394,
396

SUBPARTS 22, 23, 28, 32, 35, 173, 179, 191, 199, 236, 242, 270, 280,
305, 390, 394

SUBSET 42, 74, 93, 155, 365, 366, 390, 394

SUBSETS 42, 365, 366, 390, 394

INDEX

SUBSETTING-CRITERIA 42, 74, 93, 155, 268, 367, 390, 394
SUBSETTING-CRITERION 10, 14, 20, 62, 69, 74, 93, 155, 268, 367, 390, 394, 396
SYNONYM 6, 9, 11, 12, 14, 16, 18, 20, 22, 24, 25, 26, 28, 29, 32, 35, 38, 39, 40, 42, 43, 59, 75, 78, 94, 115, 135, 156, 180, 200, 212, 223, 243, 255, 281, 306, 323, 336, 348, 368, 384, 390, 394, 398
SYSTEM-PARAMETER 10, 62, 77, 104, 145, 165, 205, 229, 355, 390, 391, 394, 396
TERMINATED 32, 282, 391, 394
TERMINATES 8, 18, 22, 32, 50, 136, 181, 282, 283, 391, 394
TERMINATION 18, 137, 284, 391, 394
TERMINATION-CAUSES 32, 137, 284, 391, 394
TIMES-PEP 17, 21, 27, 30, 126, 169, 234, 263, 391, 394
TRACE-KEY 6, 7, 9, 10, 11, 14, 16, 18, 20, 22, 24, 25, 26, 28, 29, 32, 35, 38, 39, 40, 42, 43, 60, 62, 63, 64, 76, 95, 116, 138, 157, 182, 201, 213, 224, 244, 256, 285, 307, 324, 337, 349, 369, 385, 391, 394, 396
TRIGGERED 32, 139, 286, 391, 394
TRIGGERS 8, 18, 22, 32, 51, 139, 183, 286, 287, 391, 394
TRUE 8, 9, 17, 22, 30, 31, 32, 45, 48, 49, 50, 51, 54, 61, 120, 123, 130, 172, 265, 269, 282, 286, 391, 394
UPDATE 14, 16, 20, 22, 33, 42, 83, 96, 97, 103, 118, 144, 159, 164, 184, 196, 228, 277, 289, 302, 315, 354, 370, 371, 391, 394
UPDATED 14, 16, 20, 42, 96, 97, 117, 118, 119, 158, 159, 184, 288, 289, 370, 371, 373, 391, 394
UPDATES 32, 96, 97, 117, 118, 158, 159, 184, 288, 289, 370, 371, 391, 394
URA 1, 2, 4, 53, 68, 88, 109, 129, 150, 171, 190, 207, 219, 235, 249, 267, 297, 317, 330, 342, 359, 378, 386
URL 1, 2, 3, 4, 5, 44, 62, 78, 245, 386, 387, 392, 398, 399
USING 13, 14, 15, 16, 19, 20, 27, 30, 32, 41, 42, 85, 96, 106, 117, 147, 158, 231, 260, 288, 356, 357, 370, 391, 394
UTILIZED 33, 290, 291, 391, 394
UTILIZES 33, 290, 291, 391, 394
VALUES 11, 14, 77, 98, 391, 394
VOLATILITY 16, 119, 391, 395, 398
VOLATILITY-MEMBER 42, 372, 391, 395, 398
VOLATILITY-SET 42, 373, 391, 395, 398
WHILE 9, 61, 391, 395